

Efficient LLMs

Advances in Large Language Models

ELL8299 · AIL861, Semester 1, 2025-26



Yatin Nandwani
Research Scientist, IBM Research

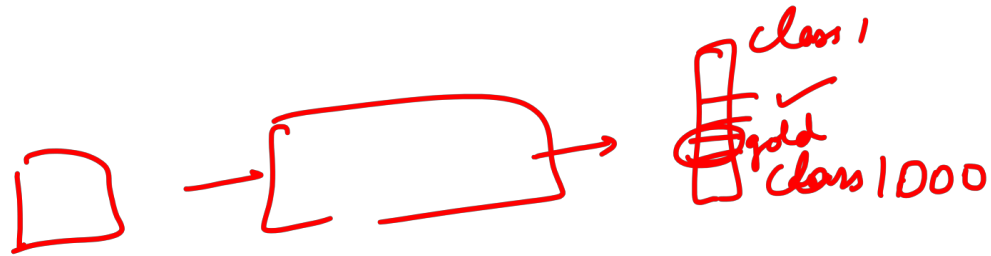
So Far...

Efficient Training

1. Parallelism for Scale -
 - *Distribute model & data to fit massive training*
 - *DP, ZeRO-1/2/3, TP (w/ SP), CP, PP*
2. Efficient Implementations
 - *Flash Attention*
 - *Liger Kernels*


Efficient Inference

How is inference different from training?

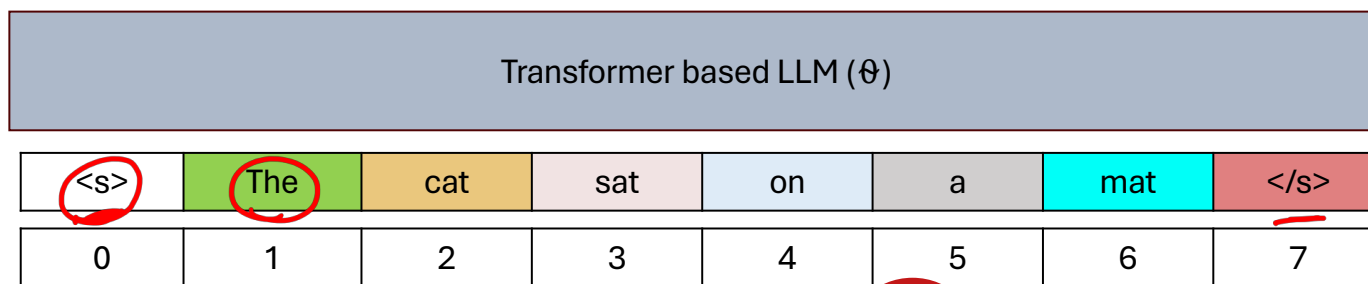


Training Vs Inference in LLMs

$$\begin{aligned}
 \text{Output} &= \text{model.forward}(X) \Rightarrow 8 \boxed{4} \\
 \text{loss} &= \text{nn.CrossEntropy}(\text{output}, \text{labels})
 \end{aligned}$$

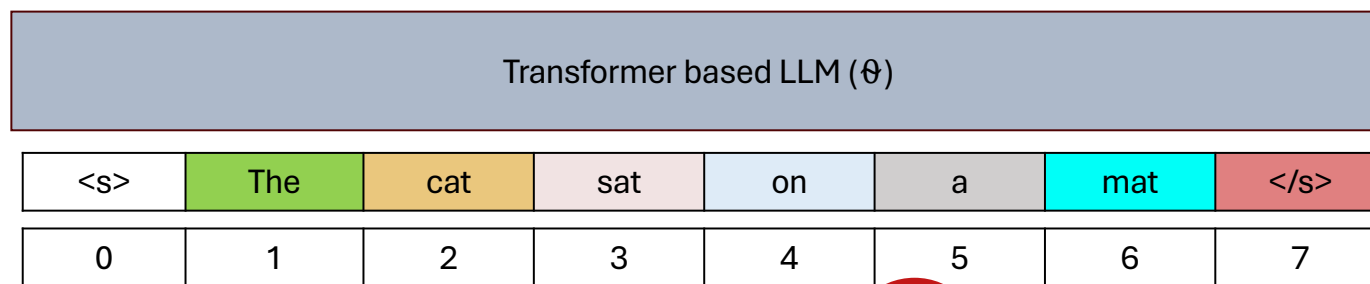
1 

Forward Pass through an LLM



Forward Pass through an LLM

Probability distribution over
all the tokens at each step
(simultaneously)



...
$p(t_1)$	$p(t_1)$					
...						
$p(t_i)$						
...
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

Forward Pass through an LLM

Probability distribution over
all the tokens at each step
(simultaneously)

Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7



...
	$p(t_1)$					
$p(The)$						

$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

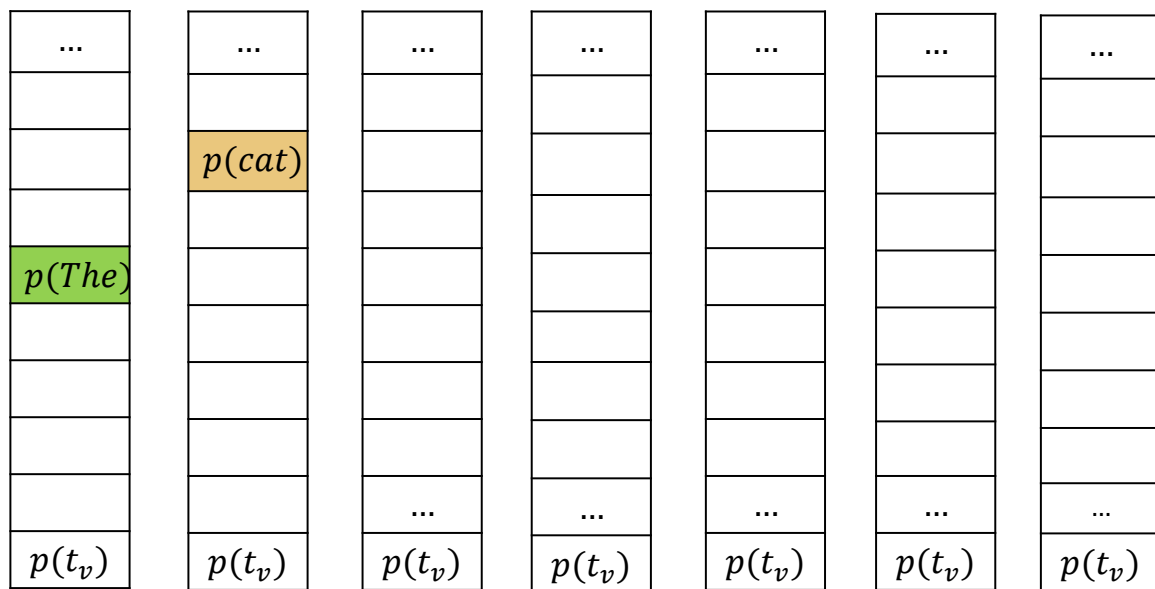
Forward Pass
through an LLM

Train to maximize prob. of
The at step 0

Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7





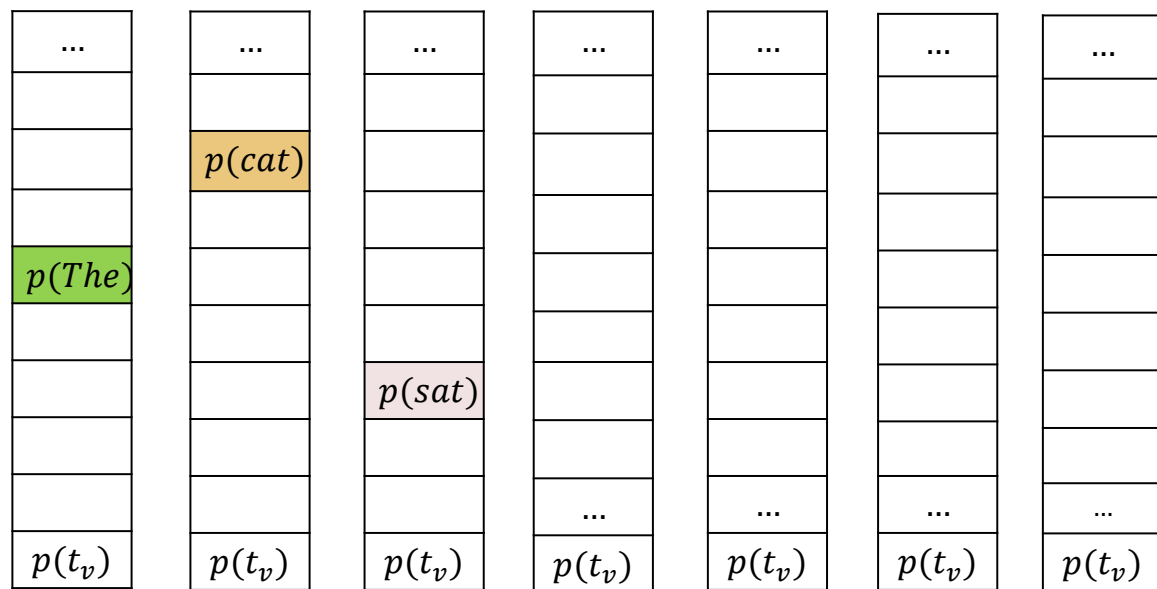
Forward Pass
through an LLM

Train to maximize prob. of
cat at step 1

Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7





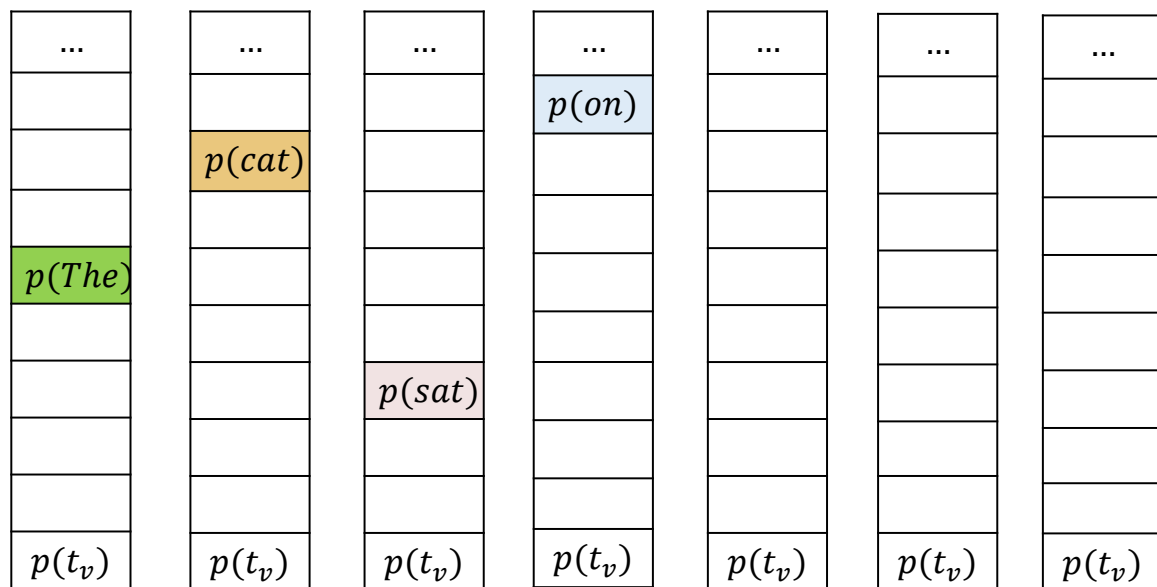
Forward Pass
through an LLM

Train to maximize prob. of
sat at step 2

Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7





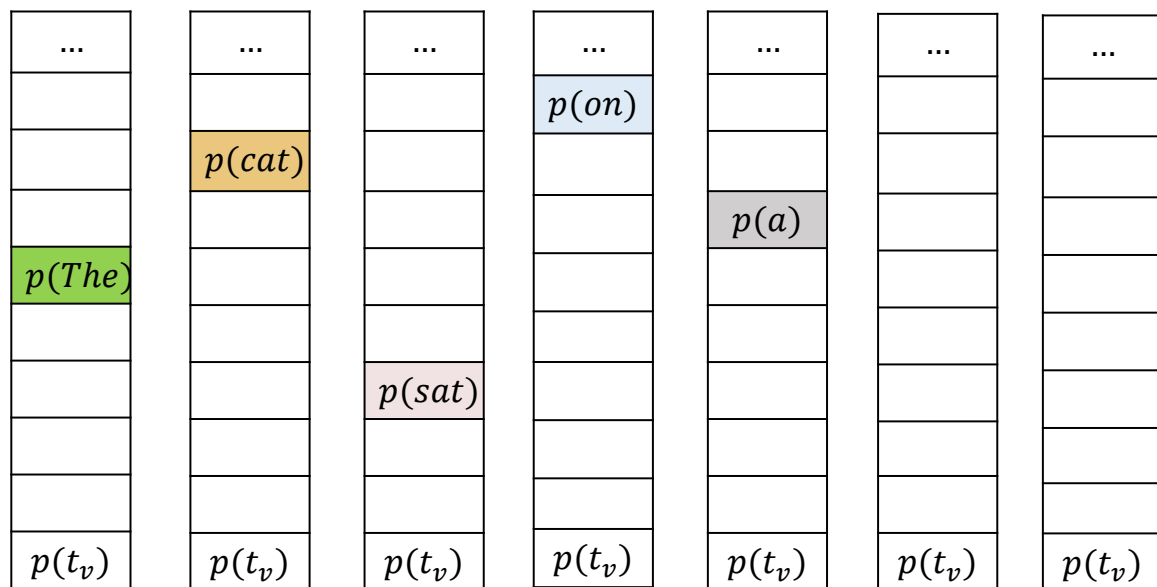
Forward Pass
through an LLM

Train to maximize prob. of
on at step 3

Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7



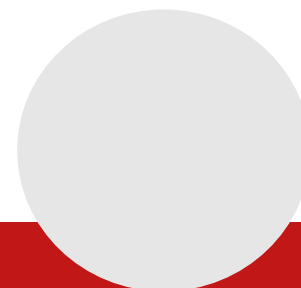


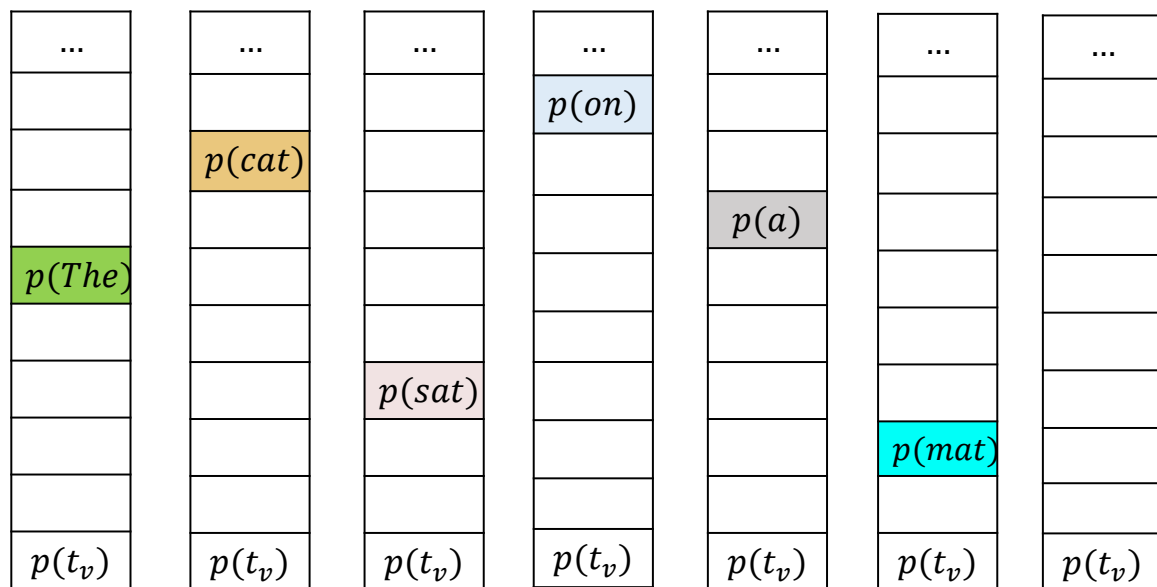
Forward Pass
through an LLM

Train to maximize prob. of
 a at step 4

Transformer based LLM (θ)

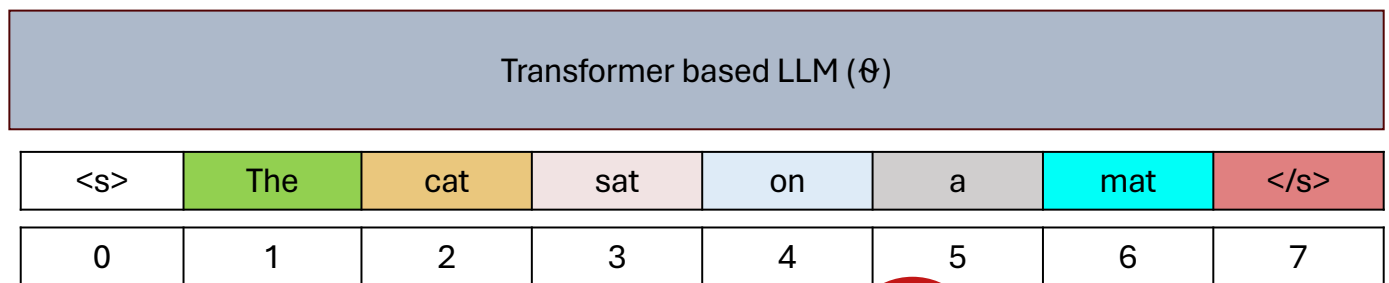
<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7

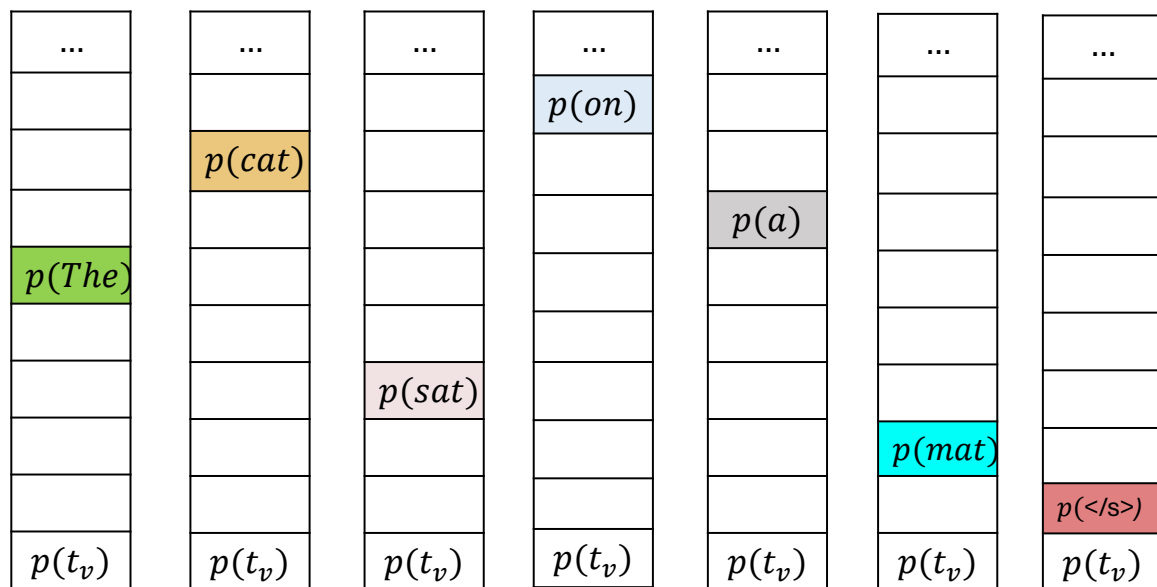




Forward Pass
through an LLM

Train to maximize prob. of
mat at step 5





Forward Pass
through an LLM

Train to maximize prob. of
 $\langle /s \rangle$ at step 6

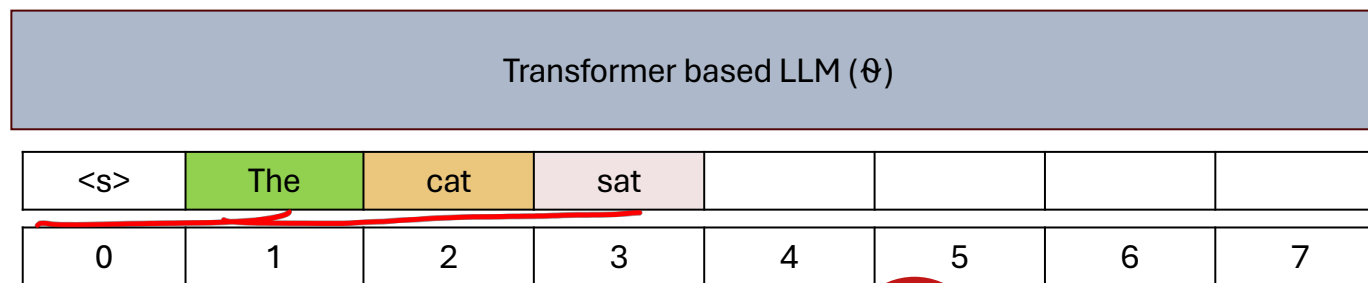
Transformer based LLM (θ)

$\langle s \rangle$	The	cat	sat	on	a	mat	$\langle /s \rangle$
0	1	2	3	4	5	6	7



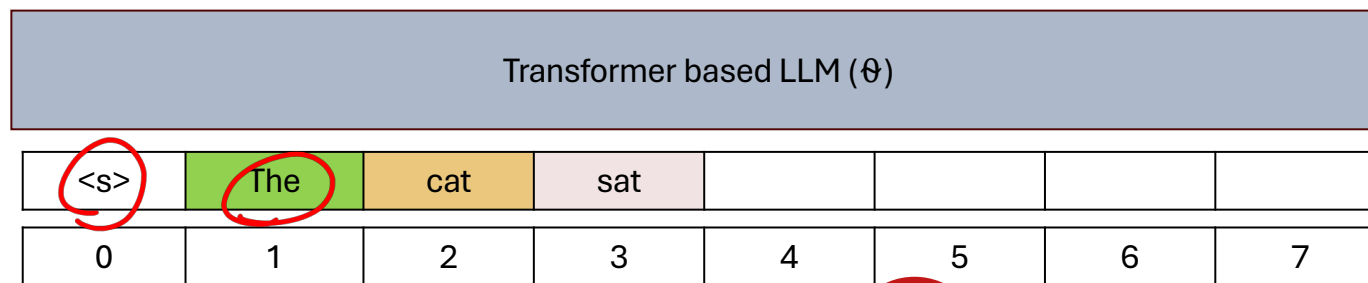
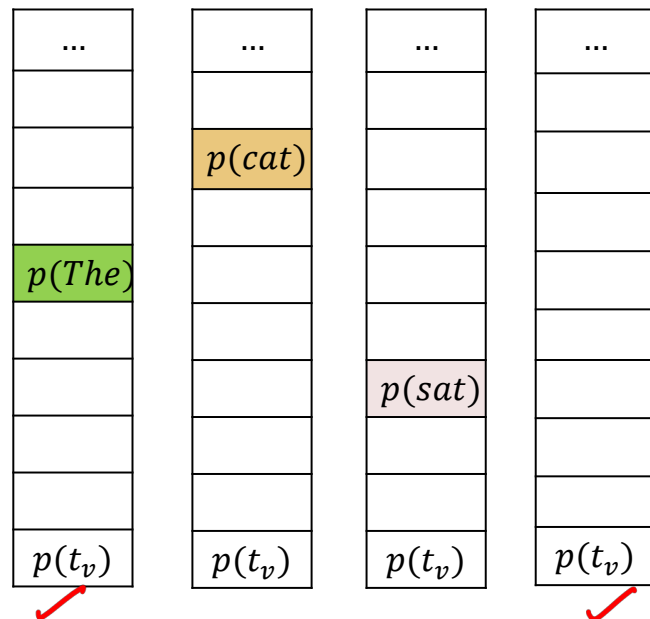
Inference through an LLM

Forward Pass (#1)



Inference through an LLM

Prob. Dist. at all steps



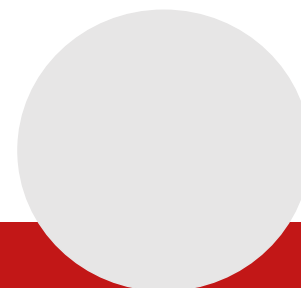
...
	$p(cat)$		
$p(The)$			
		$p(sat)$	
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

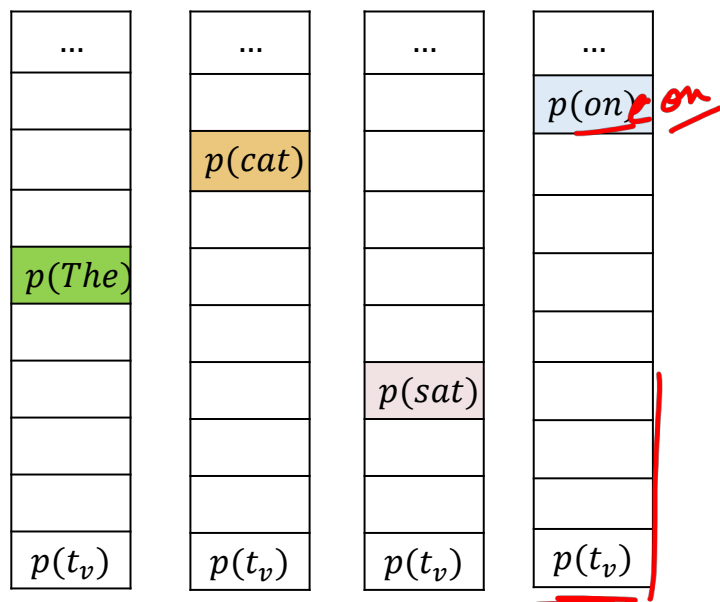
Inference through an LLM

Pick the token having max. probability at step 3

Transformer based LLM (θ)

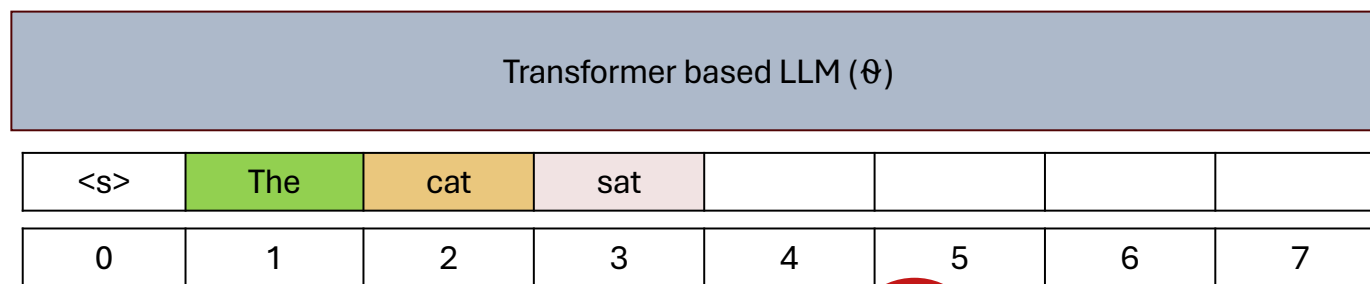
<s>	The	cat	sat				
0	1	2	3	4	5	6	7

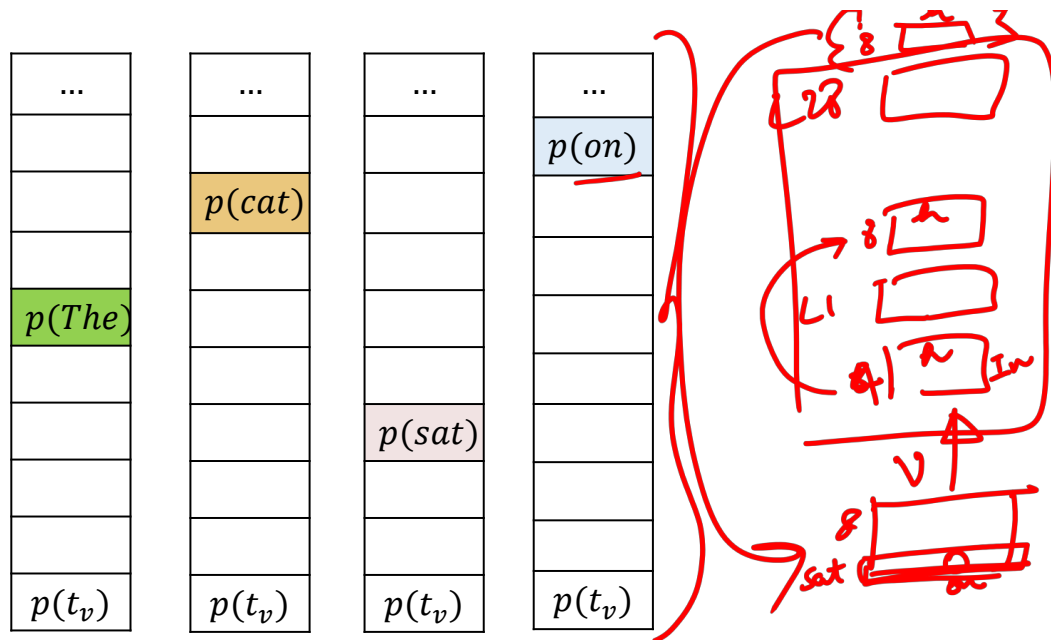




Inference through an LLM

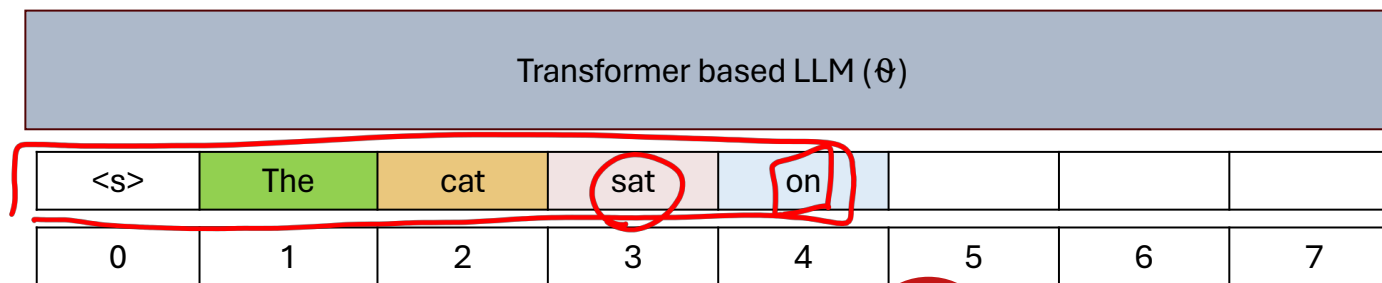
Pick the token having max. probability at step 3





Inference through
an LLM

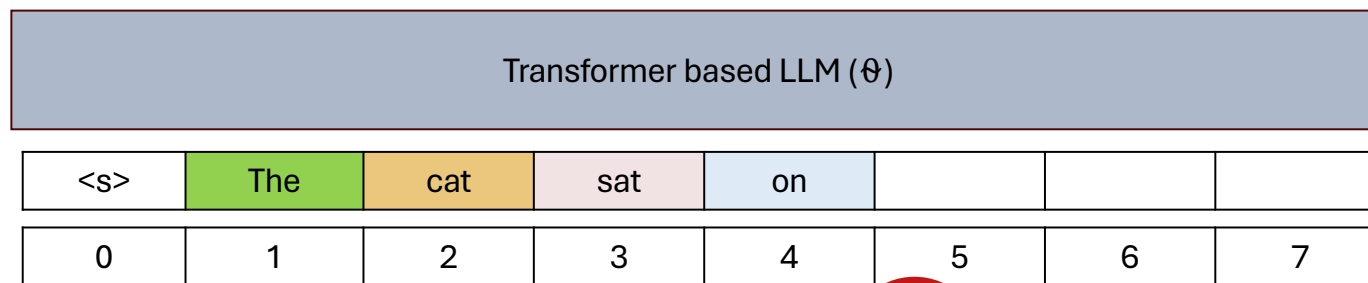
Fill at step 4



Inference through an LLM

Fwd. Pass (#2)

5 ↩



...
			$p(on)$	
	$p(cat)$			
				$p(a)$
$p(The)$				
		$p(sat)$		
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

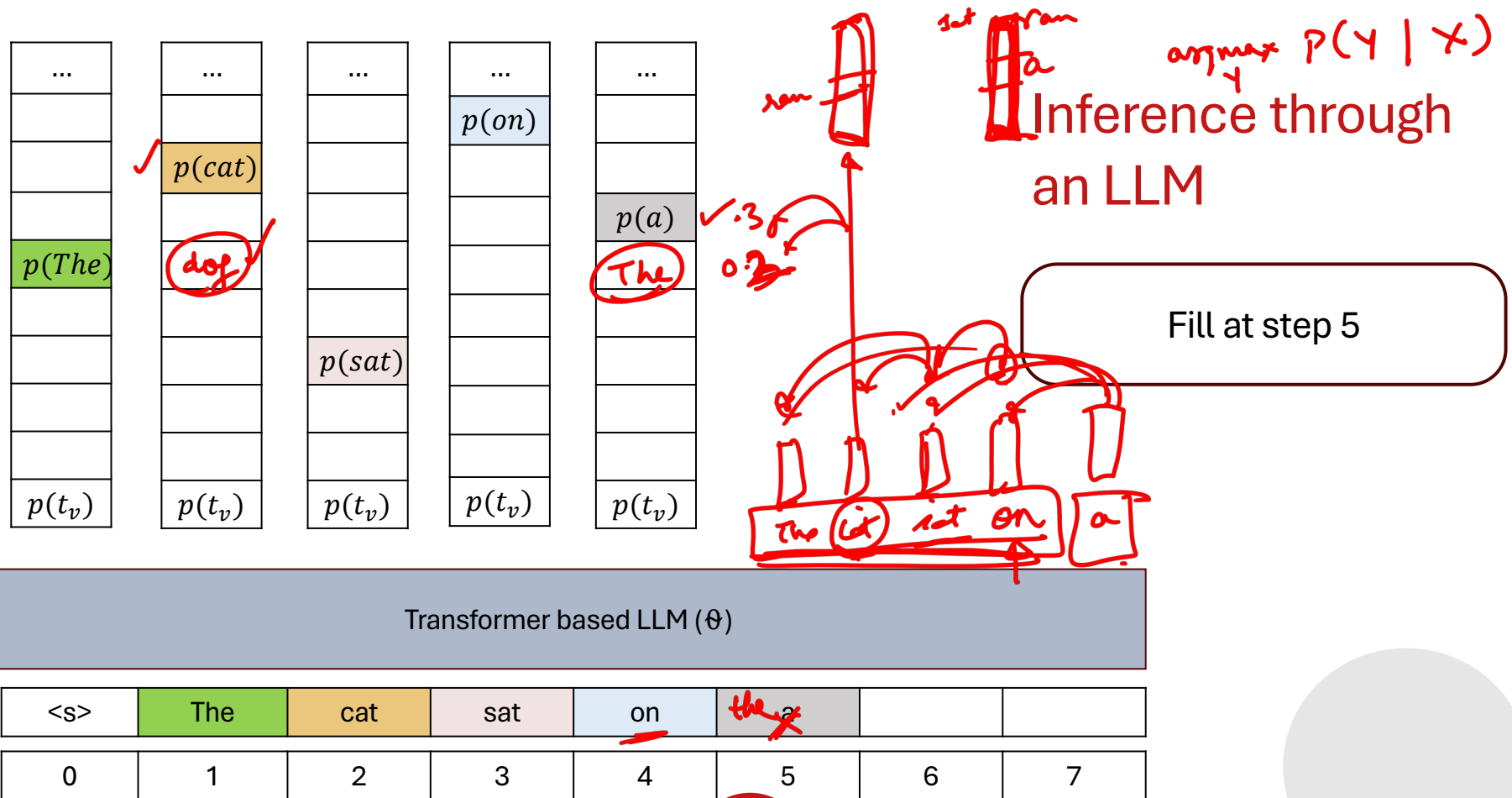
Inference through an LLM

Fwd. pass (#2) to get distribution at step 4

Transformer based LLM (θ)

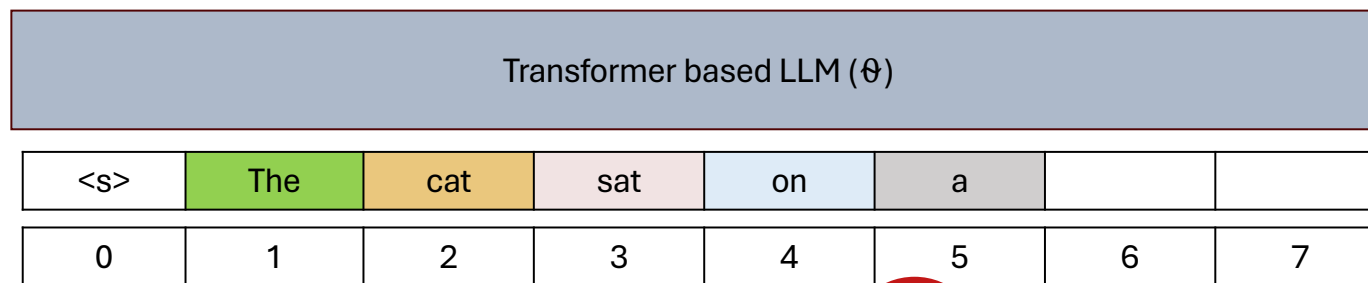
<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7





Inference through an LLM

Fwd. pass again (#3)



...
			$p(on)$		
	$p(cat)$				
				$p(a)$	
$p(The)$					
		$p(sat)$			
					$p(mat)$
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

Inference through an LLM

Fwd. pass again (#3)

Transformer based LLM (θ)

<s>	The	cat	sat	on	a		
0	1	2	3	4	5	6	7



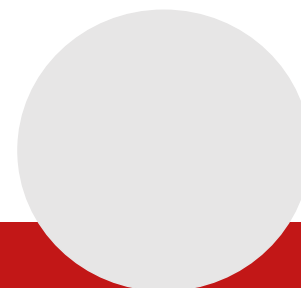
...
			$p(on)$		
	$p(cat)$				
				$p(a)$	
$p(The)$					
		$p(sat)$			
					$p(mat)$
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

Inference through an LLM

Fill at step 6

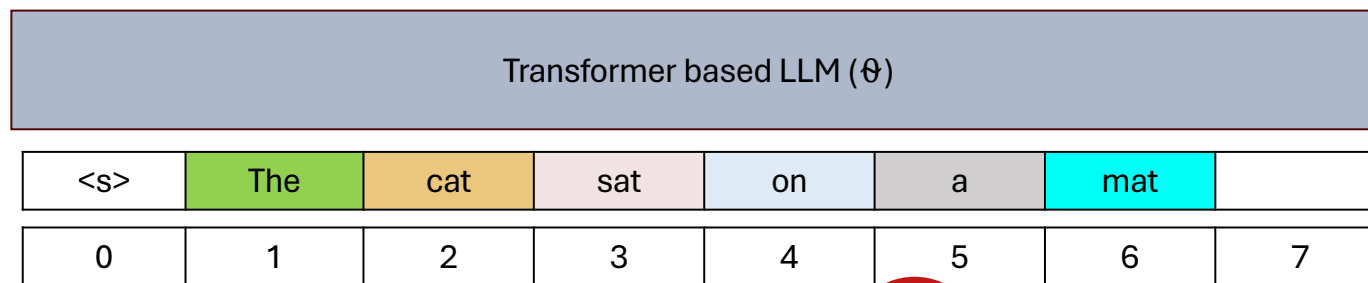
Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	
0	1	2	3	4	5	6	7



Inference through an LLM

Fwd. pass again (#4)



✗ 10 10 10

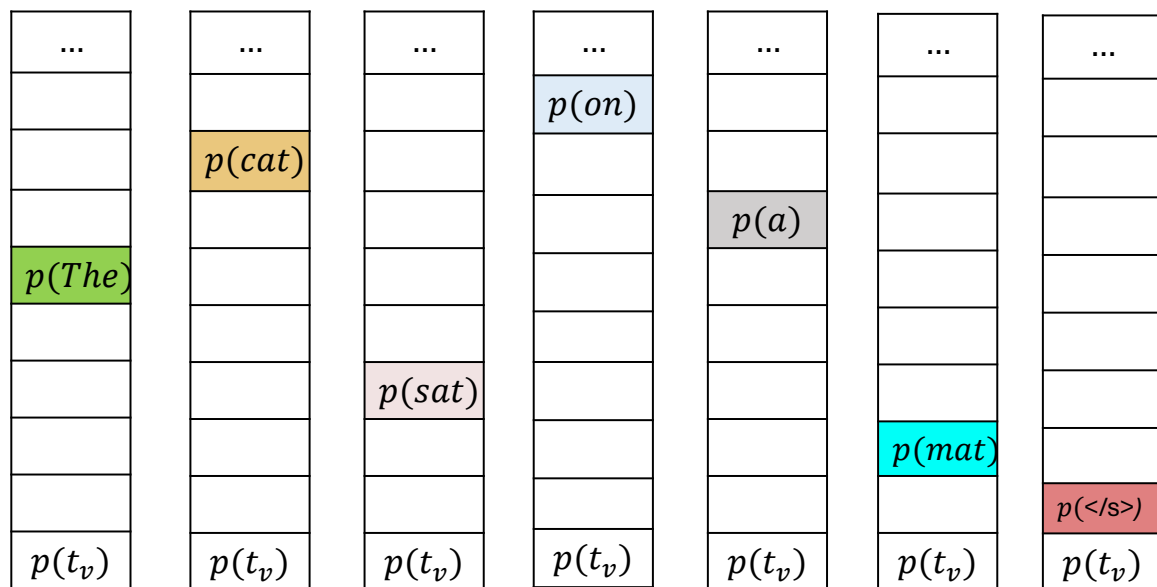
Inference through an LLM

Fwd. pass again (#4)

...
			$p(on)$			
	$p(cat)$			$p(a)$		
$p(The)$						
		$p(sat)$				
					$p(mat)$	
						$p(</s>)$
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

Transformer based LLM (θ)							
<s>	The	cat	sat	on	a	mat	
0	1	2	3	4	5	6	7



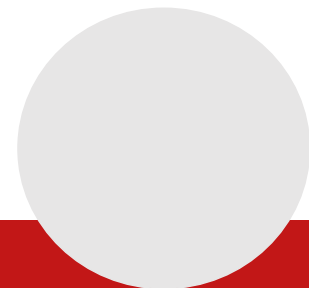


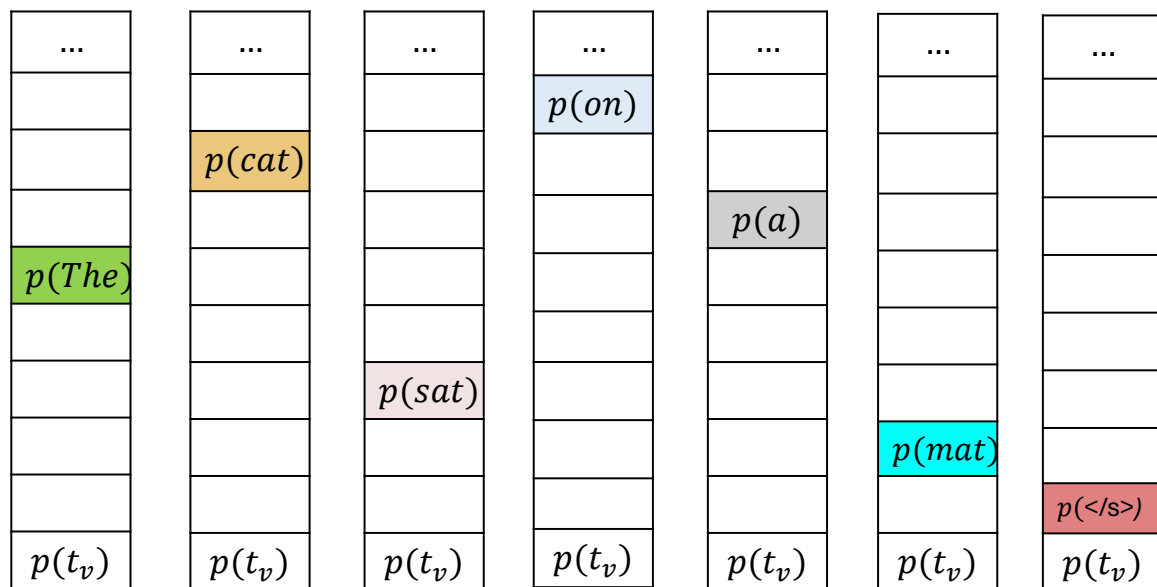
Inference through an LLM

Stop at end of seq. token:
 $</s>$

Transformer based LLM (θ)

$<s>$	The	cat	sat	on	a	mat	$</s>$
0	1	2	3	4	5	6	7



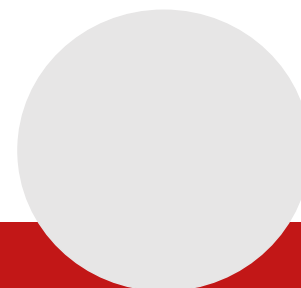


Inference through
an LLM

Fwd Passes: 4
#Tokens: 4

Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7



Inference through an LLM

- ☐ 4 forward passes for 4 tokens
- ☐ Not feasible at production scale
- ☐ Let us revisit forward pass through and see if we can optimize
- ☐ We will focus on attention layer as that is the bottleneck

Fwd Passes: 4
#Tokens: 4

Transformer based LLM (θ)

<s>	The	cat	sat	on	a	mat	</s>
0	1	2	3	4	5	6	7



Why we need efficient inference?

Training

- Single forward pass and all output probabilities computed in parallel

Inference

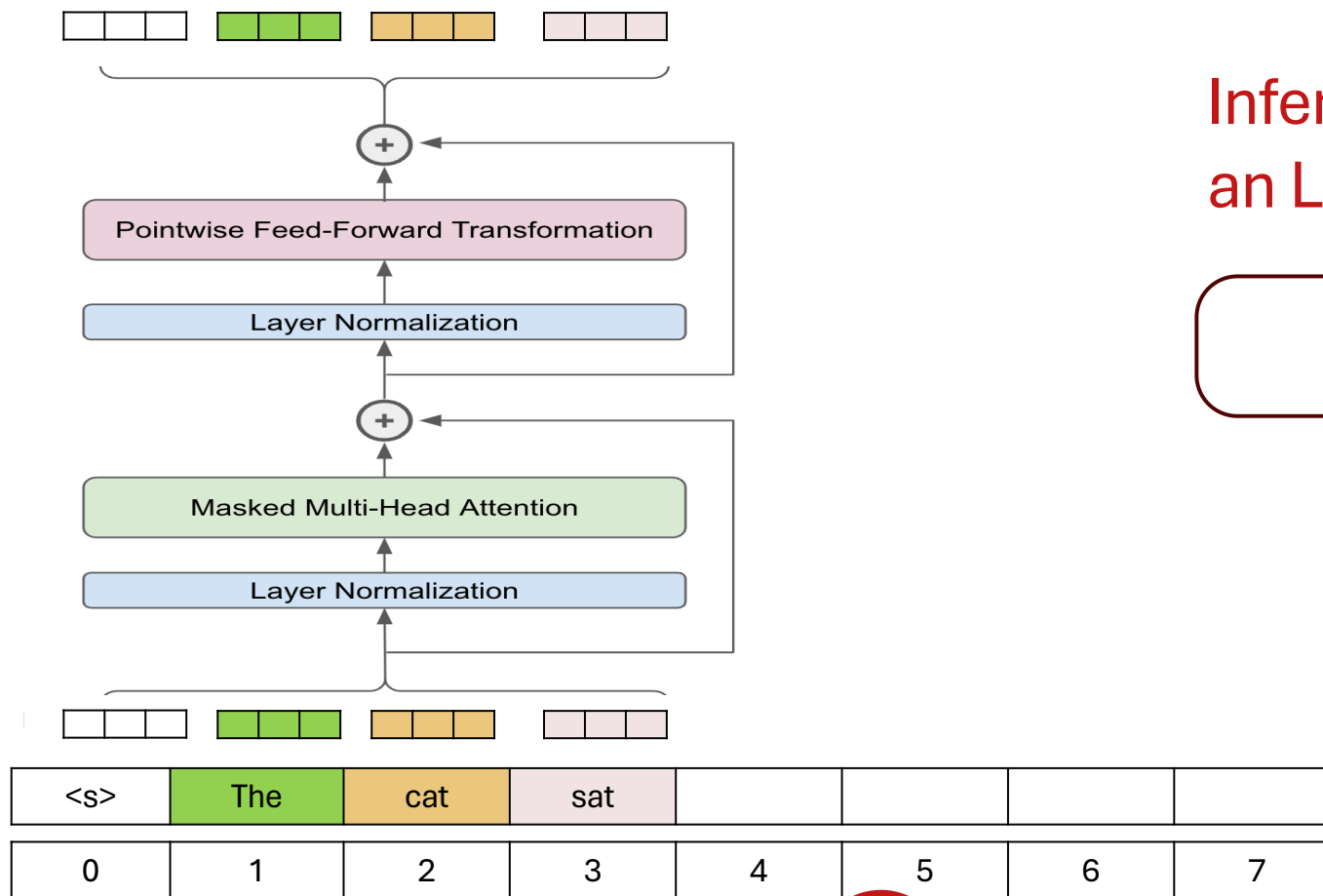
- One forward pass for each token 😞
- Very expensive
- Need techniques to make it workable

Are there any redundant computations in each iteration?



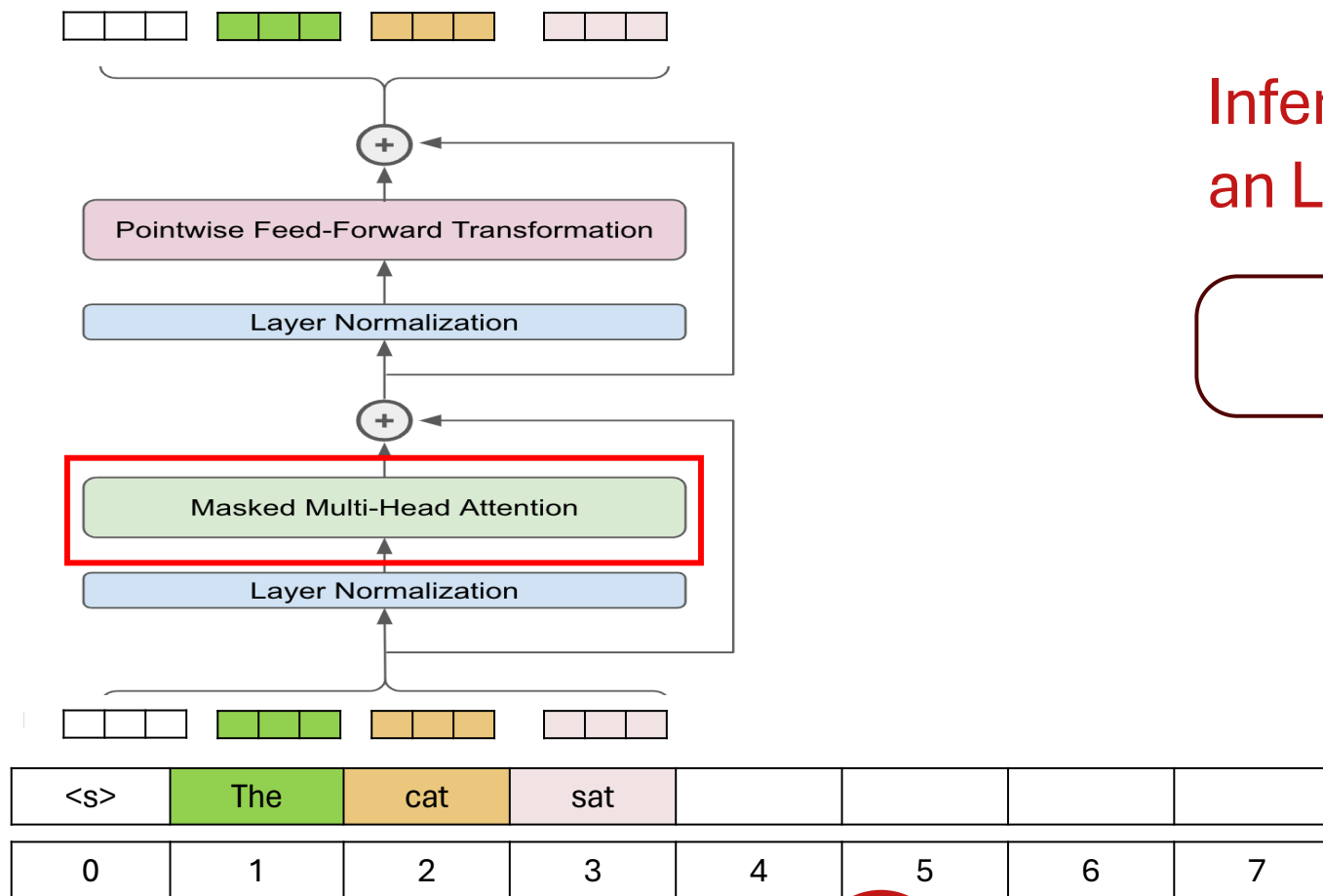
Inference through an LLM

Forward Pass #1



Inference through an LLM

Forward Pass #1

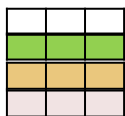


Inference through an LLM

Forward Pass #1

<s>	The	cat	sat				
0	1	2	3	4	5	6	7





Inference through an LLM

Forward Pass #1

<s>	The	cat	sat				
0	1	2	3	4	5	6	7

Content credits: <https://cameronrwolfe.substack.com/p/decoder-only-transformers-the-workhorse>



W_Q



<s>
The
cat
sat

Q: $4 \times d$ dim.

W_K



<s>
The
cat
sat

K: $4 \times d$ dim.

W_V



<s>
The
cat
sat

V: $4 \times d$ dim.

Inference through an LLM

Forward Pass #1

<s>	The	cat	sat				
0	1	2	3	4	5	6	7



Inference through an LLM

Forward Pass #1

$Q: 4 \times d$ dim.

<s>
The
cat
sat

$V: 4 \times d$ dim.

<s>
The
cat
sat

<s>	The	cat	sat
-----	-----	-----	-----

$K^T: d \times 4$ dim.

<s>	The	cat	sat				
0	1	2	3	4	5	6	7



Inference through an LLM

Forward Pass #1

Q: $4 \times d$ dim.

<s>
The
cat
sat

A: 4×4 dim.

$$A = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)$$

V: $4 \times d$ dim.

<s>
The
cat
sat

<s>	The	cat	sat
-----	-----	-----	-----

K^T: $d \times 4$ dim.

<s>	The	cat	sat				
0	1	2	3	4	5	6	7



Inference through an LLM

Q: $4 \times d$ dim.

<s>
The
cat
sat

A: 4×4 dim.

1			
0.2	0.8		
0.1	0.3	0.6	
0.01	0.19	0.3	0.5

V: $4 \times d$ dim.

<s>
The
cat
sat

<s>	The	cat	sat
-----	-----	-----	-----

K^T: $d \times 4$ dim.

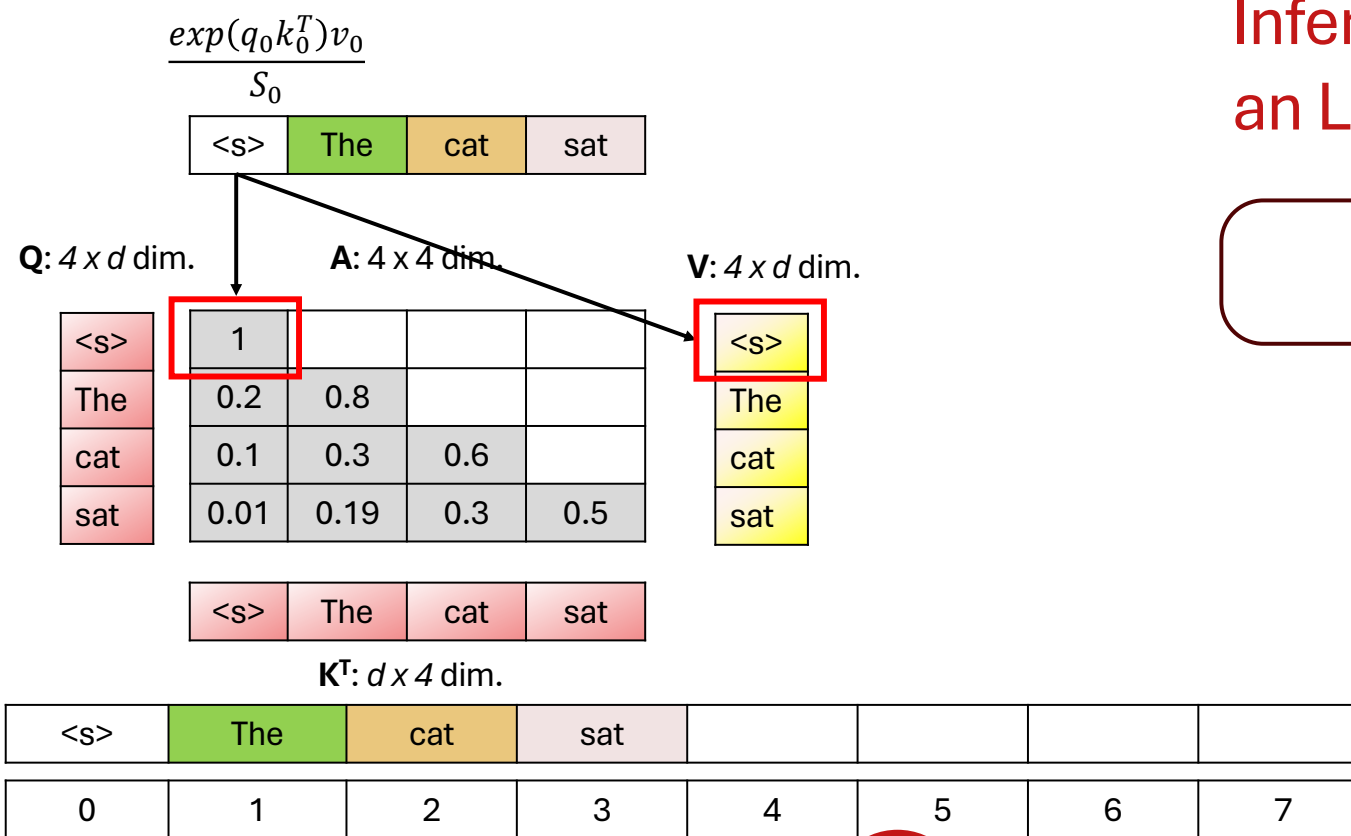
<s>	The	cat	sat				
0	1	2	3	4	5	6	7

Forward Pass #1



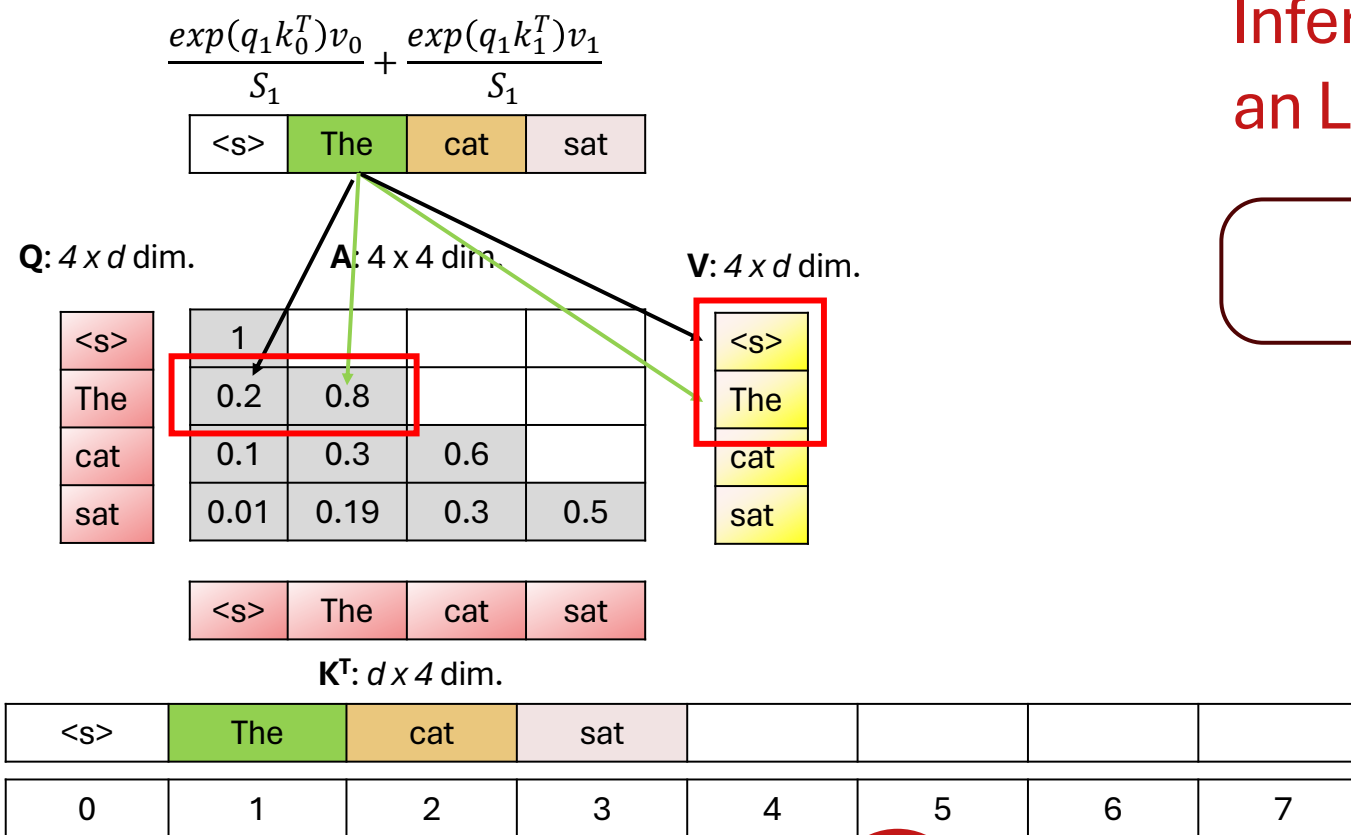
Inference through an LLM

Forward Pass #1



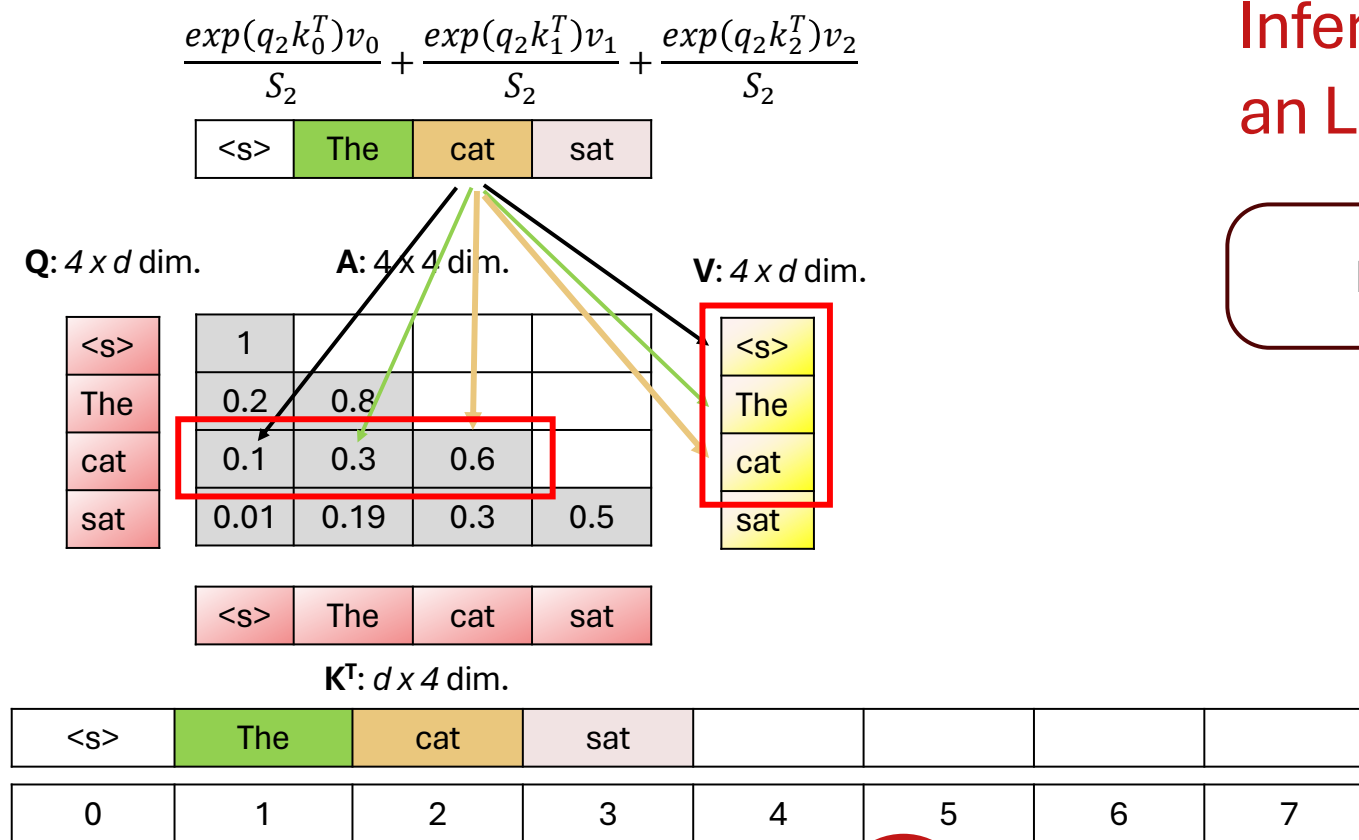
Inference through an LLM

Forward Pass #1



Inference through an LLM

Forward Pass #1



$$\frac{\exp(q_3 k_0^T) v_0}{S_3} + \frac{\exp(q_3 k_1^T) v_1}{S_3} + \frac{\exp(q_3 k_2^T) v_2}{S_3} + \frac{\exp(q_3 k_3^T) v_3}{S_3}$$

<s> The cat sat

Q: $4 \times d$ dim.

A: 4×4 dim

V: $4 \times d$ dim.

<s>
The
cat
sat

1			
0.2	0.8		
0.1	0.3	0.6	
0.01	0.19	0.3	0.5

<s>
The
cat
sat

<s> The cat sat

K^T: $d \times 4$ dim.

<s>	The	cat	sat				
0	1	2	3	4	5	6	7

Inference through an LLM

Forward Pass #1



...
	$p(cat)$		
$p(The)$			
		$p(sat)$	
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

Inference through an LLM

- Emb. of **sat** at the last layer
- Pass through classifier to get distribution over tokens
- Pick the token having max. probability at step 3

Transformer based LLM (θ)

<s>	The	cat	sat				
0	1	2	3	4	5	6	7



...
			$p(\text{on})$
	$p(\text{cat})$		
$p(\text{The})$			
		$p(\text{sat})$	
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

Inference through an LLM

- Emb. of **sat** at the last layer
- Pass through classifier to get distribution over tokens
- Pick the token having max. probability at step 3

Transformer based LLM (θ)

<s>	The	cat	sat				
0	1	2	3	4	5	6	7



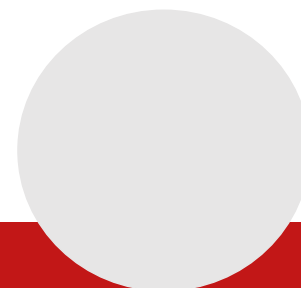
...
			$p(on)$
	$p(cat)$		
$p(The)$			
		$p(sat)$	
$p(t_v)$	$p(t_v)$	$p(t_v)$	$p(t_v)$

Inference through an LLM

Fill at step 4

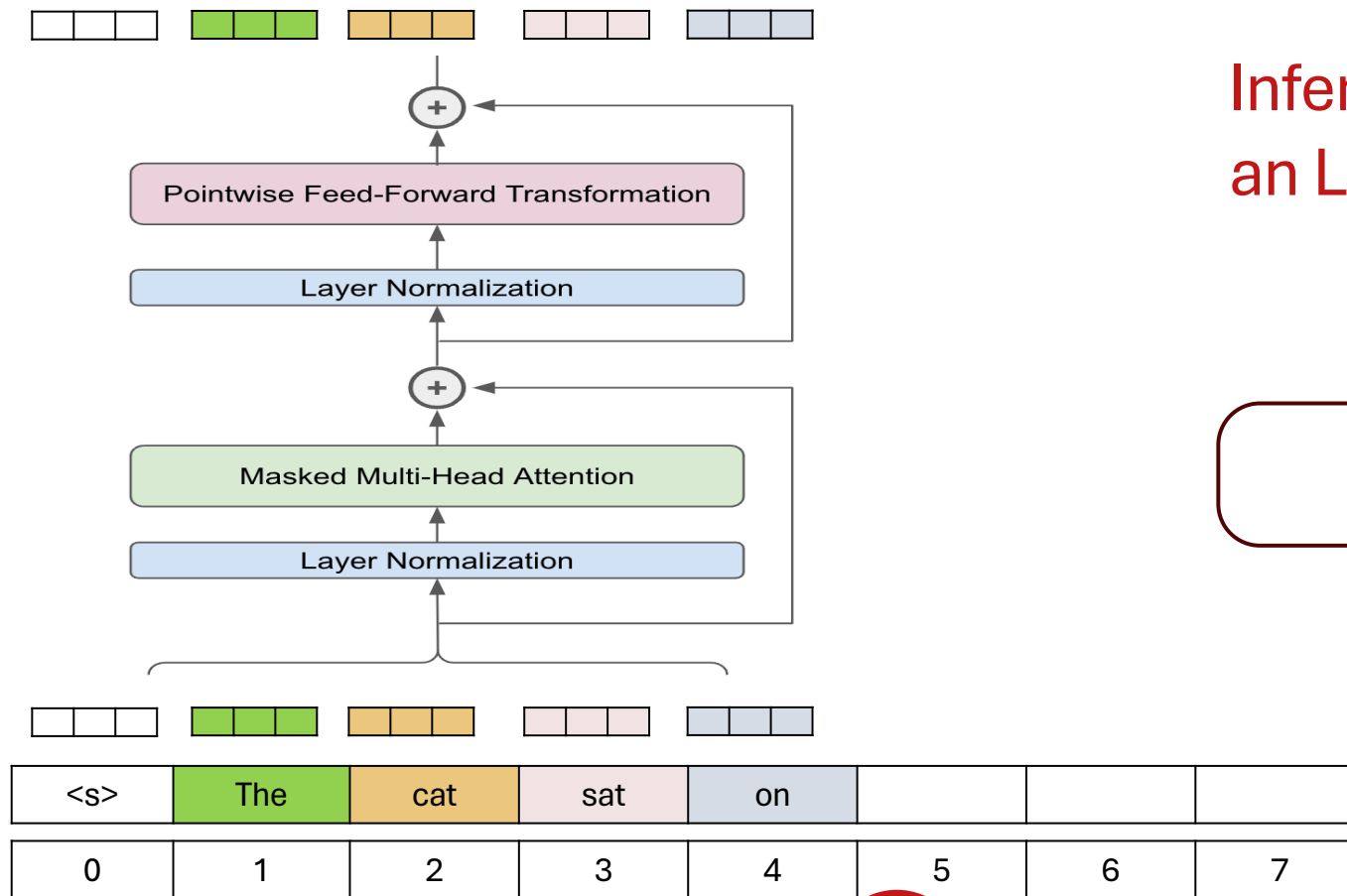
Transformer based LLM (θ)

<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7



Inference through an LLM

Forward Pass #2



Content credits: <https://cameronwolfe.substack.com/p/decoder-only-transformers-the-workhorse>



Inference through an LLM

Q: $5 \times d$ dim.

A: 5×5 dim.

V: $5 \times d$ dim.

<s>
The
cat
sat
on

$$A = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)$$

<s>
The
cat
sat
on

<s>	The	cat	sat	on
-----	-----	-----	-----	----

K^T: $d \times 5$ dim.

<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7

- A lot of computation already done in Fwd. pass #1

Forward Pass #2



Inference through an LLM

- Attention matrix already computed in #1

Q: $5 \times d$ dim.

A: 5×5 dim.

V: $5 \times d$ dim.

<s>	1			
The	0.2	0.8		
cat	0.1	0.3	0.6	
sat	0.01	0.19	0.3	0.5
on				

<s>
The
cat
sat
on

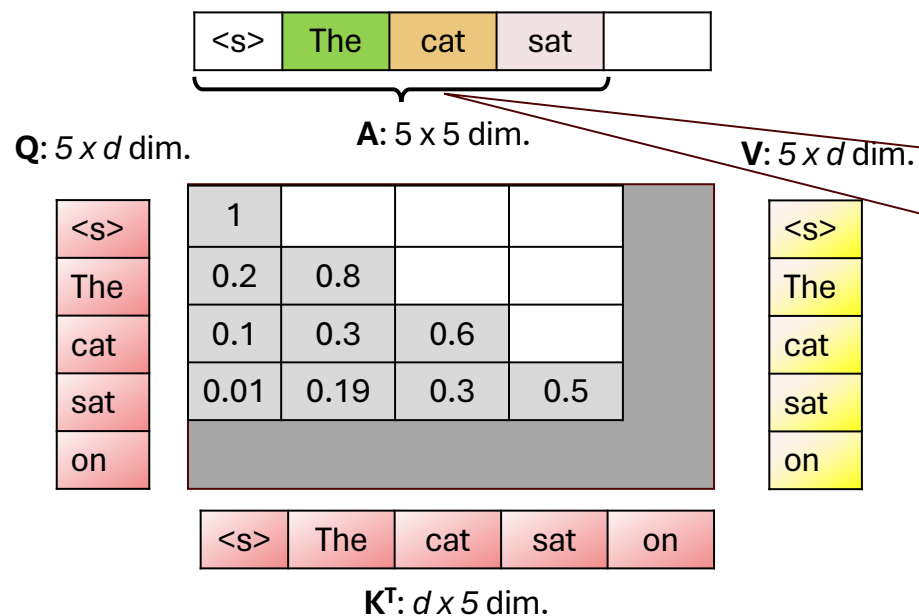
<s>	The	cat	sat	on
-----	-----	-----	-----	----

~~**K:** $d \times 5$ dim.~~

<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7



Inference through an LLM

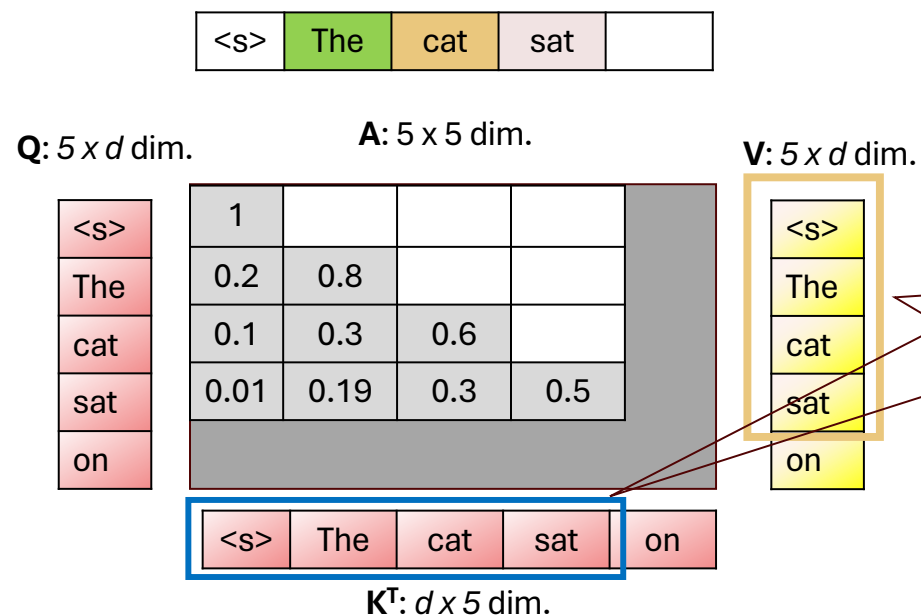


- Attention matrix already computed in #1
- Output embed. already computed in #1

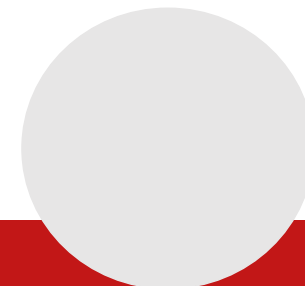
$\langle s \rangle$	The	cat	sat	on			
0	1	2	3	4	5	6	7



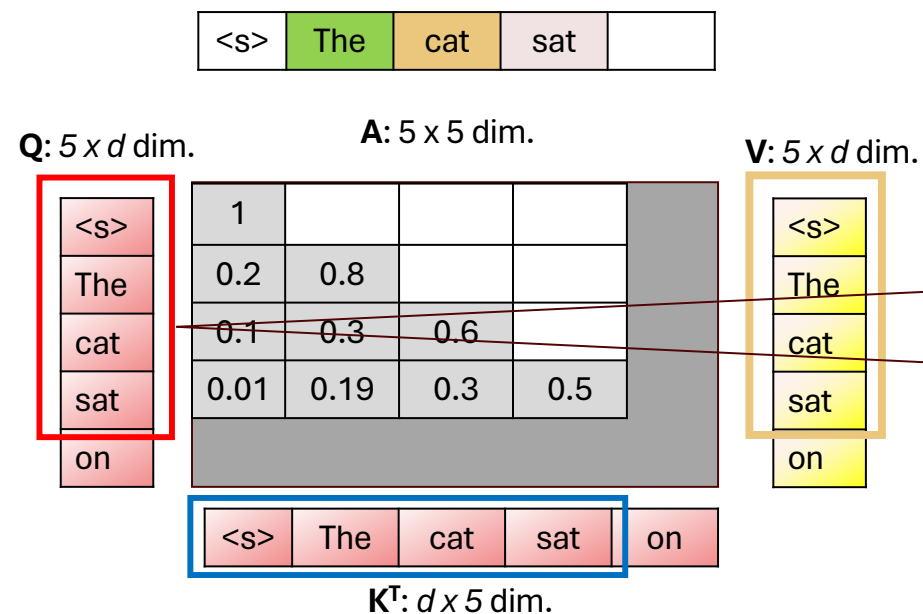
Inference through an LLM



- Attention matrix already computed in #1
- Output embed. already computed in #1
- Keys and Values already computed in #1



Inference through an LLM



- Attention matrix already computed in #1
- Output embed. already computed in #1
- Keys and Values already computed in #1
- Queries not required in #2

$\langle s \rangle$	The	cat	sat	on			
0	1	2	3	4	5	6	7



Inference through an LLM

<s>	The	cat	sat	
-----	-----	-----	-----	--

Q: $5 \times d$ dim.

A: 5×5 dim.

V: $5 \times d$ dim.

<s>	1			
The	0.2	0.8		
cat	0.1	0.3	0.6	
sat	0.01	0.19	0.3	0.5
on				

<s>
The
cat
sat
on

<s>	The	cat	sat	on
-----	-----	-----	-----	----

K^T: $d \times 5$ dim.

<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7

- Cache the already computed matrices



Inference through an LLM

<s> The cat sat

$Q: 5 \times d$ dim.

$A: 5 \times 5$ dim.

$V: 5 \times d$ dim.

<s>
The
cat
sat
on

1			
0.2	0.8		
0.1	0.3	0.6	
0.01	0.19	0.3	0.5

<s>
The
cat
sat
on

K cache
<s>
The
cat
sat

<s> The cat sat on

$K^T: d \times 5$ dim.

<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7



Inference through an LLM

<s> The cat sat

$Q: 5 \times d$ dim.

$A: 5 \times 5$ dim.

$V: 5 \times d$ dim.

<s>
The
cat
sat
on

1			
0.2	0.8		
0.1	0.3	0.6	
0.01	0.19	0.3	0.5

<s>
The
cat
sat
on

K cache
<s>
The
cat
sat

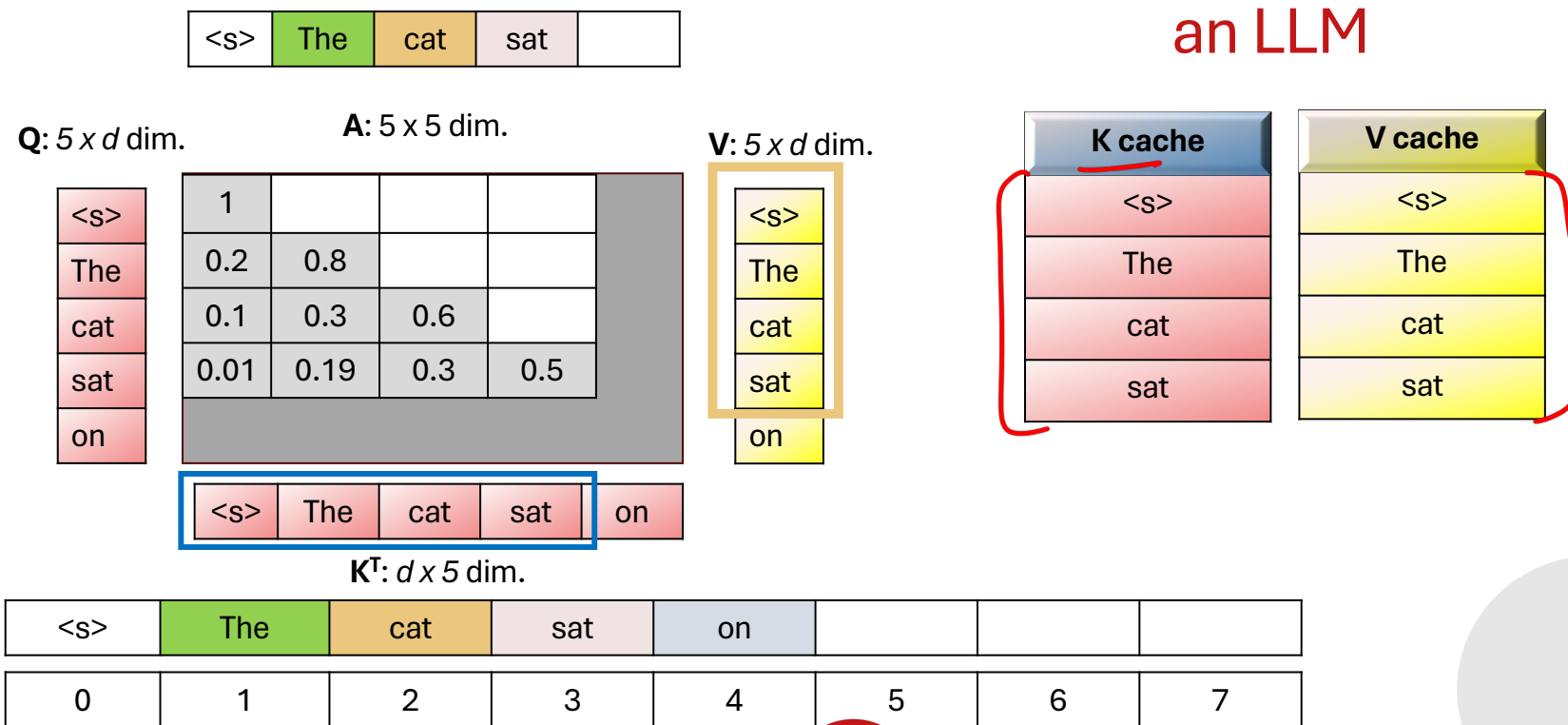
<s> The cat sat on

$K^T: d \times 5$ dim.

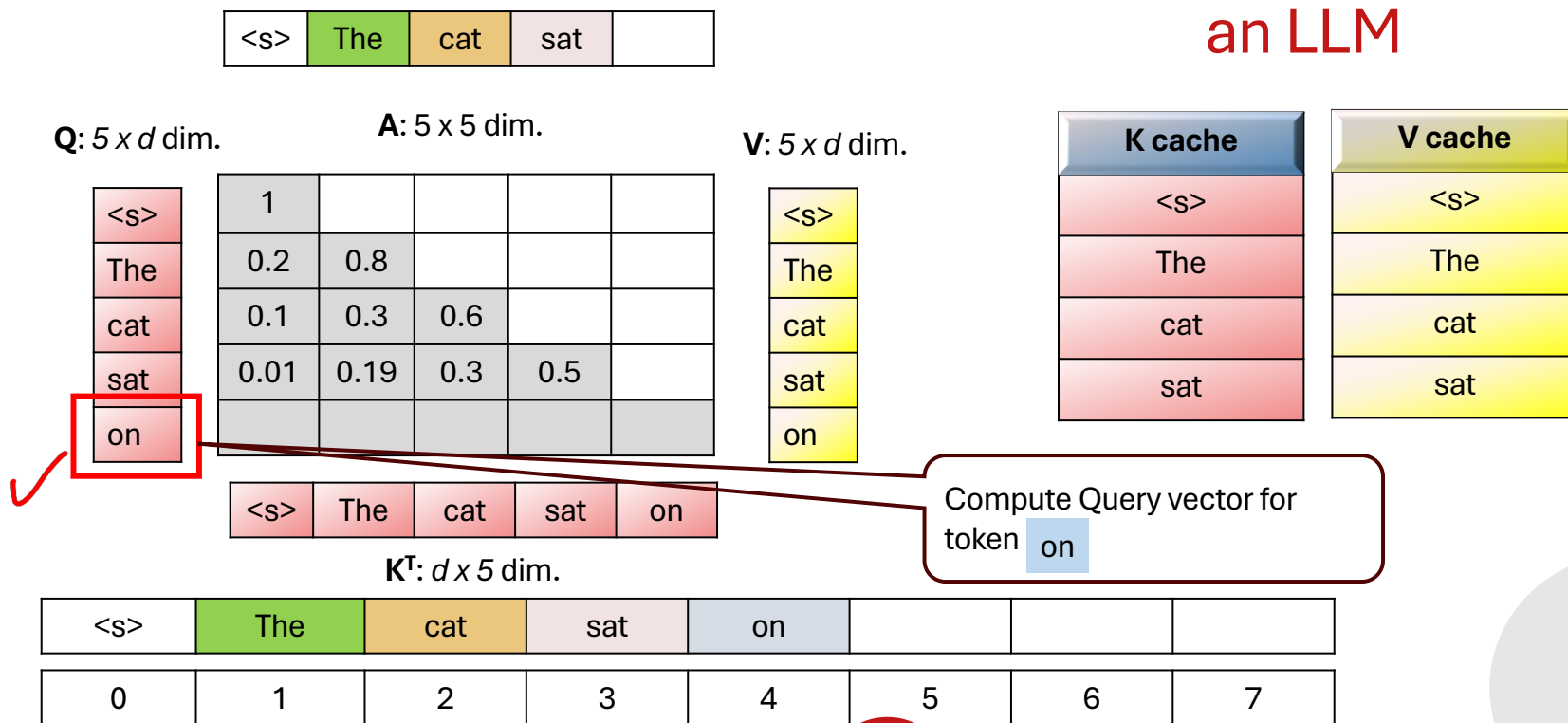
<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7



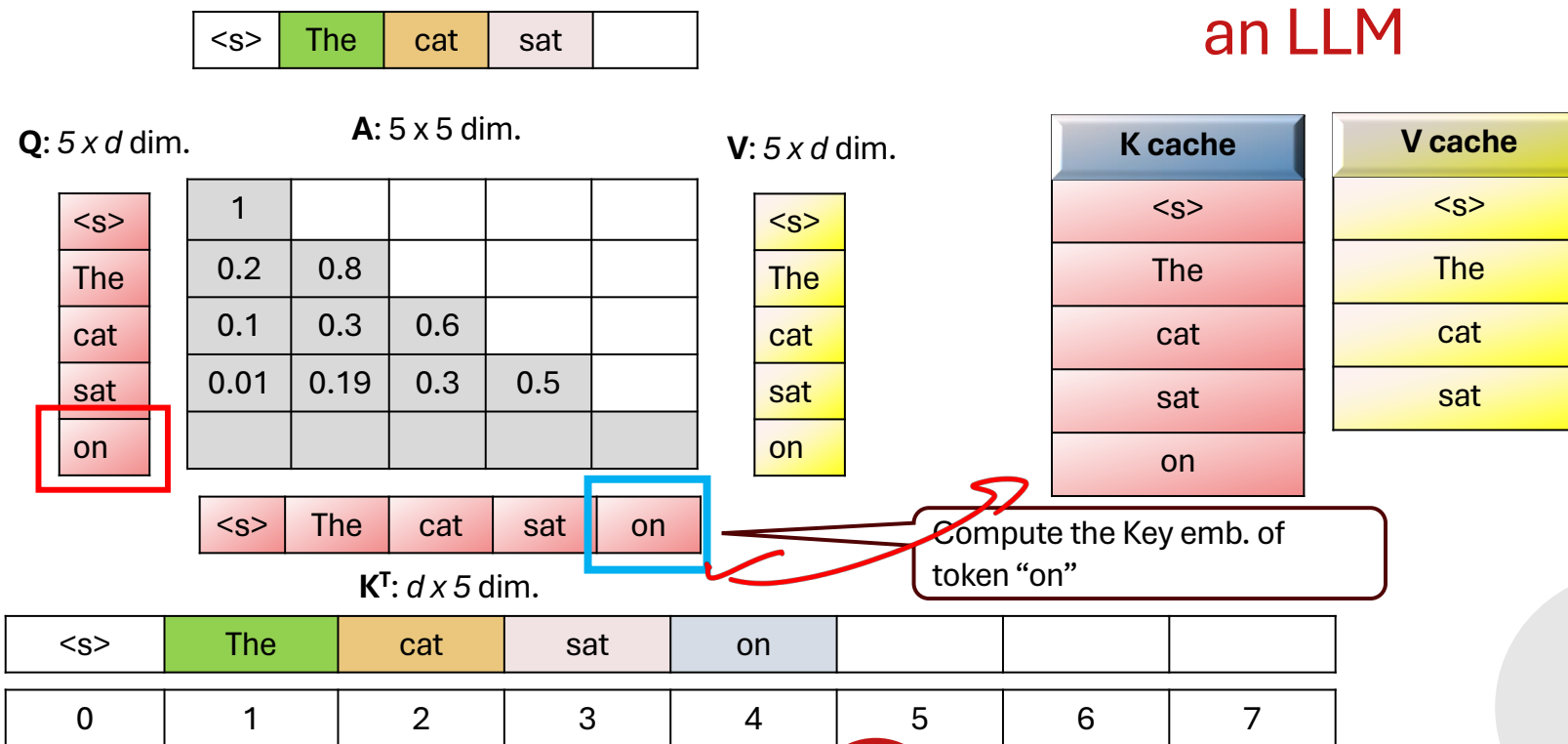
Inference through an LLM



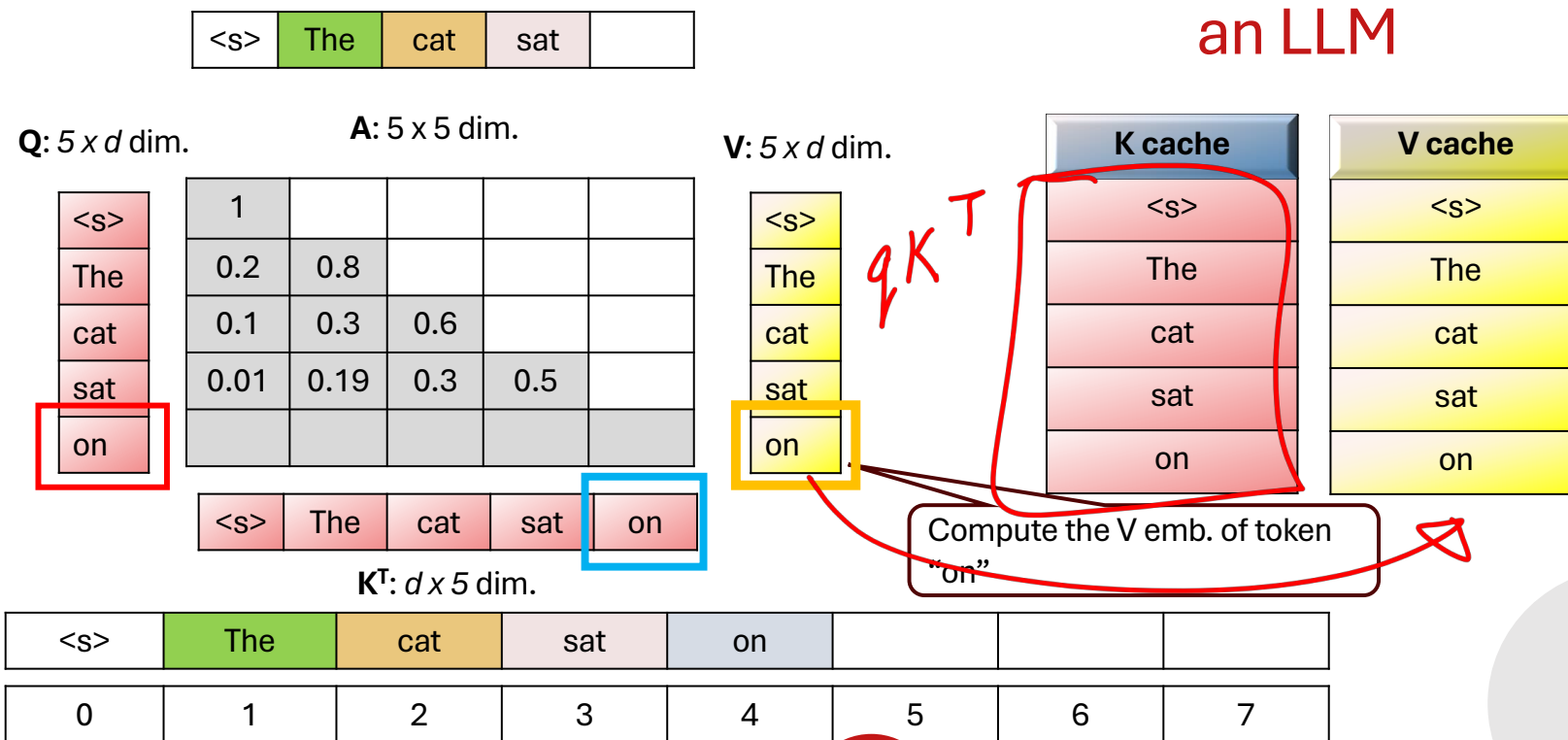
Inference through an LLM



Inference through an LLM

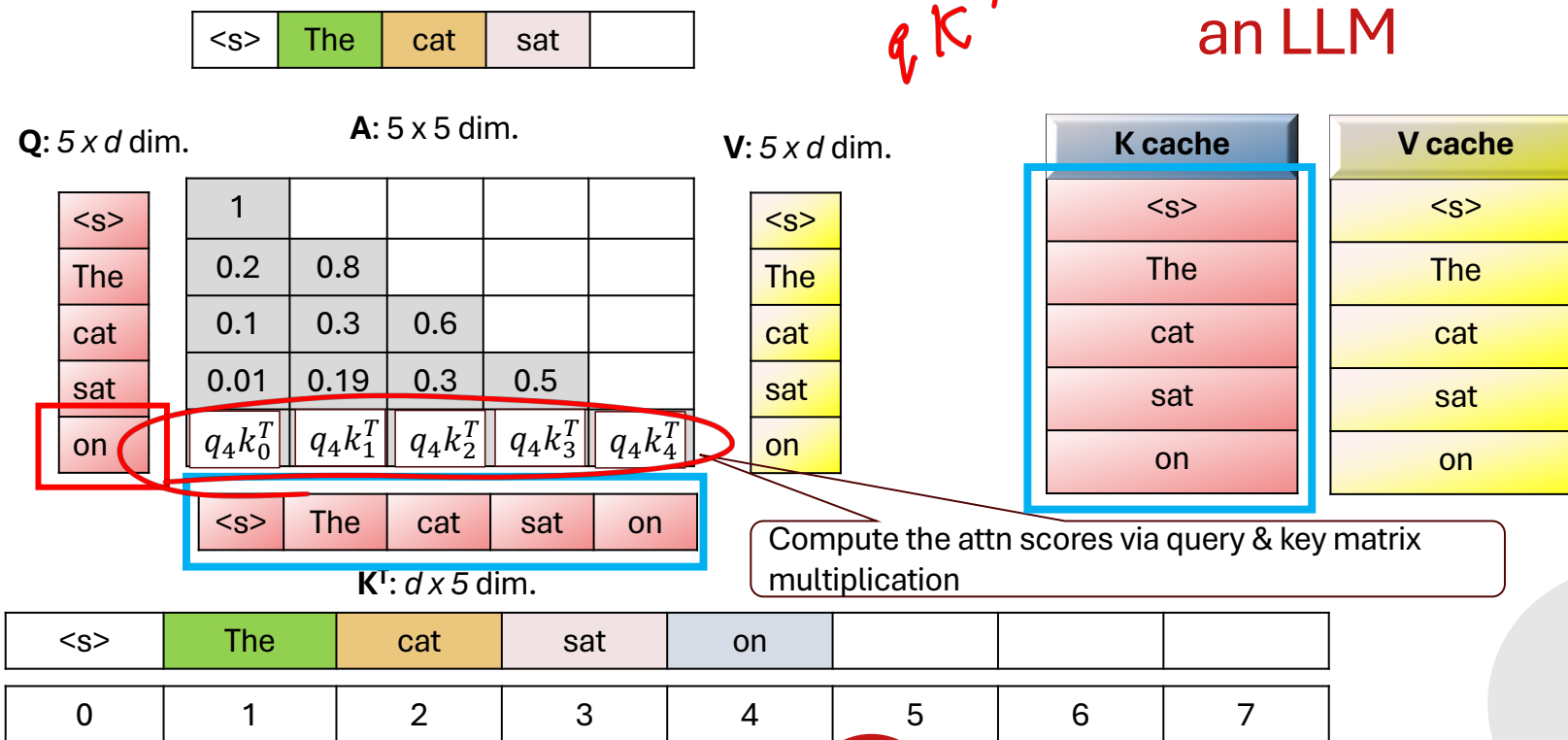


Inference through an LLM

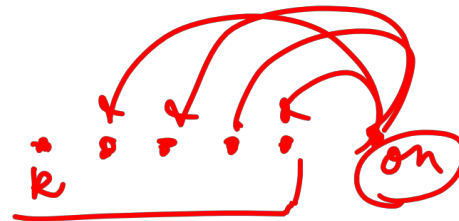


Inference through an LLM

q, k^T



<s> The cat sat



Inference through an LLM

Q: $5 \times d$ dim.

A: 5×5 dim.

<s>	1				
The	0.2	0.8			
cat	0.1	0.3	0.6		
sat	0.01	0.19	0.3	0.5	
on	0.03	0.07	0.1	0.3	0.4

<s> The cat sat on

K^T : $d \times 5$ dim.

<s>	The	cat	sat	on			
0	1	2	3	4	5	6	7

V: $5 \times d$ dim.

<s>
The
cat
sat
on

K cache
<s>
The
cat
sat
on

V cache
<s>
The
cat
sat
on

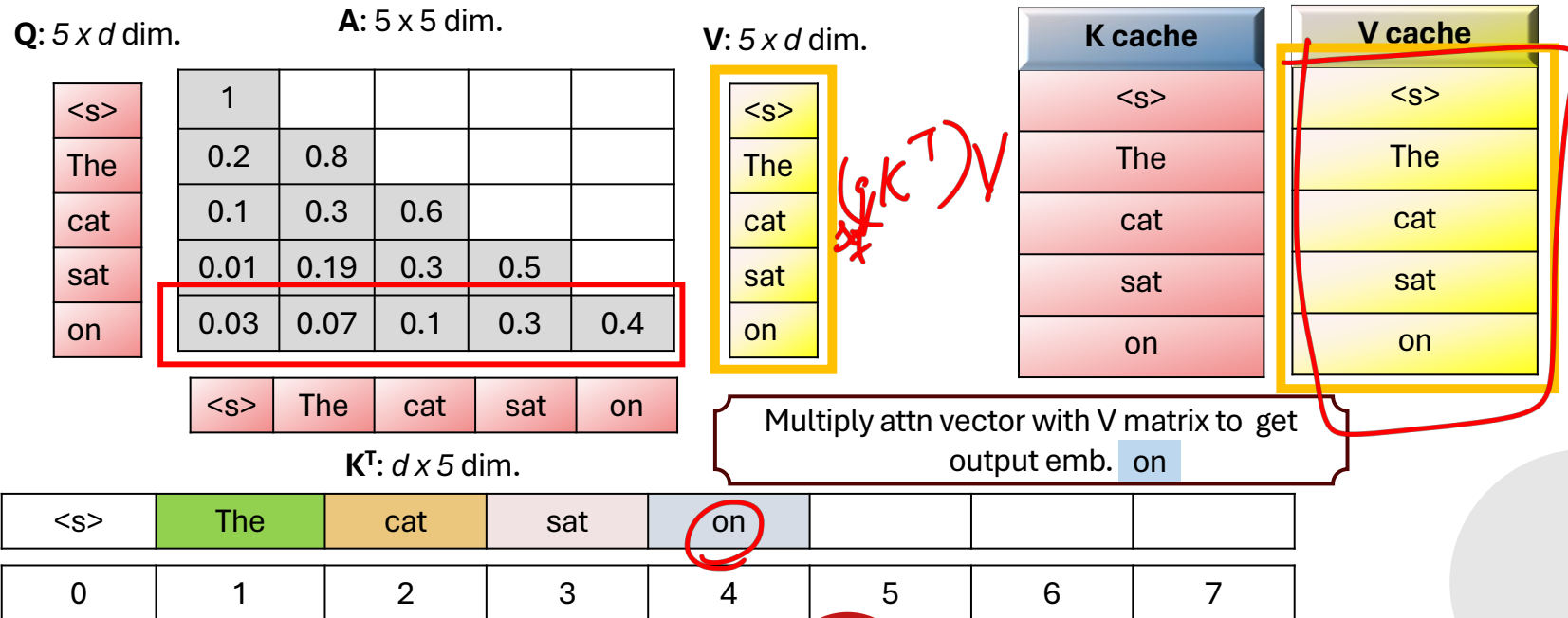
Softmax: Convert attn. scores to probability

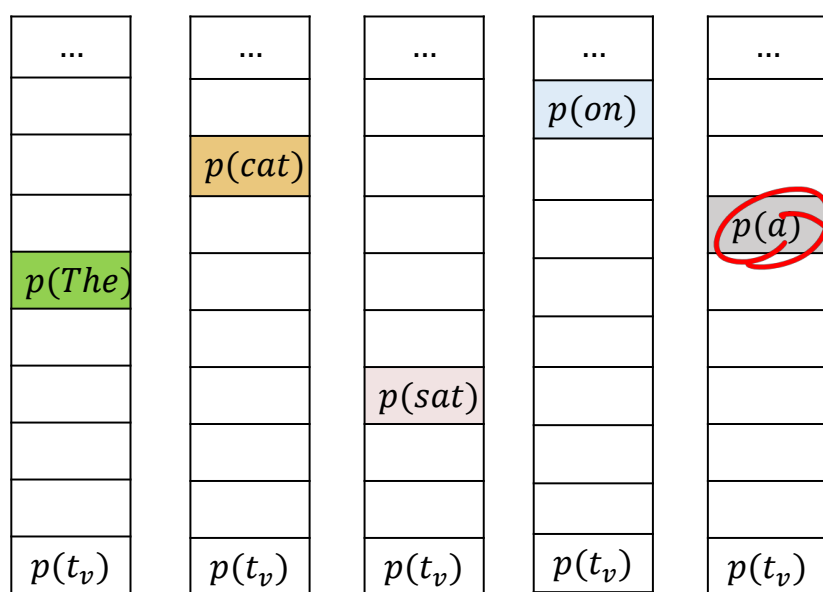


$$\frac{(q_4 k_0^T) v_0}{S_4} + \frac{(q_4 k_1^T) v_1}{S_4} + \frac{(q_4 k_2^T) v_2}{S_4} + \frac{(q_4 k_3^T) v_3}{S_4} + \frac{(q_4 k_4^T) v_4}{S_4}$$

<s>	The	cat	sat	on
-----	-----	-----	-----	----

Inference through an LLM





Inference through an LLM

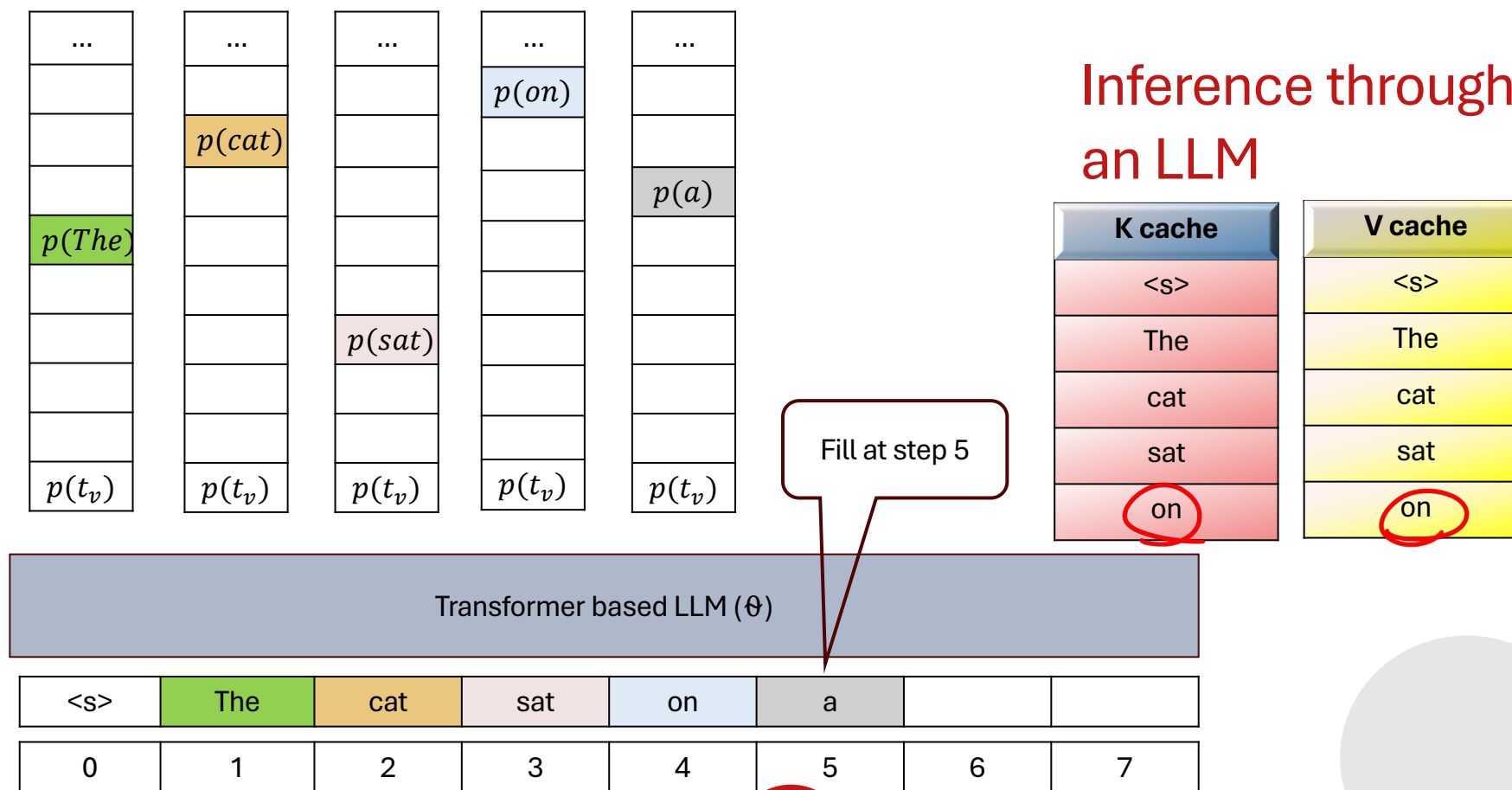
K cache	V cache
<s>	<s>
The	The
cat	cat
sat	sat
on	on

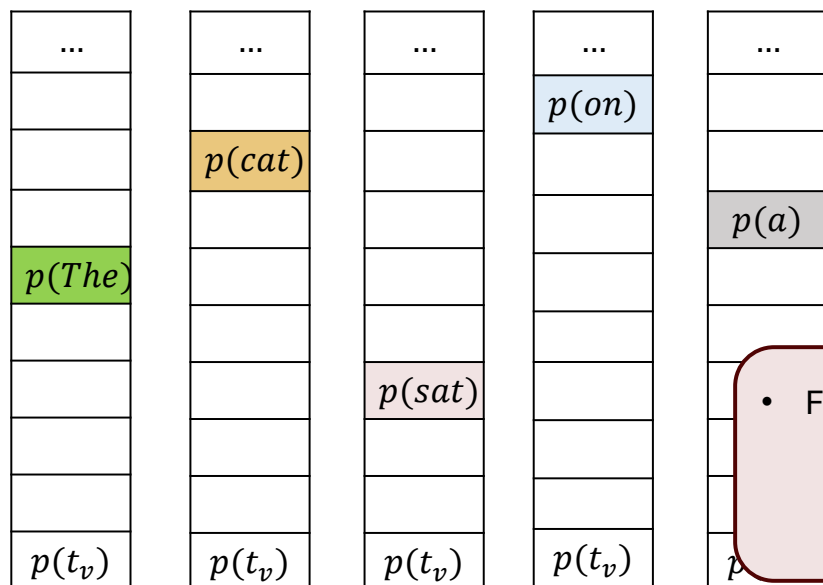
Transformer based LLM (θ)

<s>	The	cat	sat	on	a		
0	1	2	3	4	5	6	7



Inference through an LLM





- Fwd. pass again (#3)

Inference through an LLM

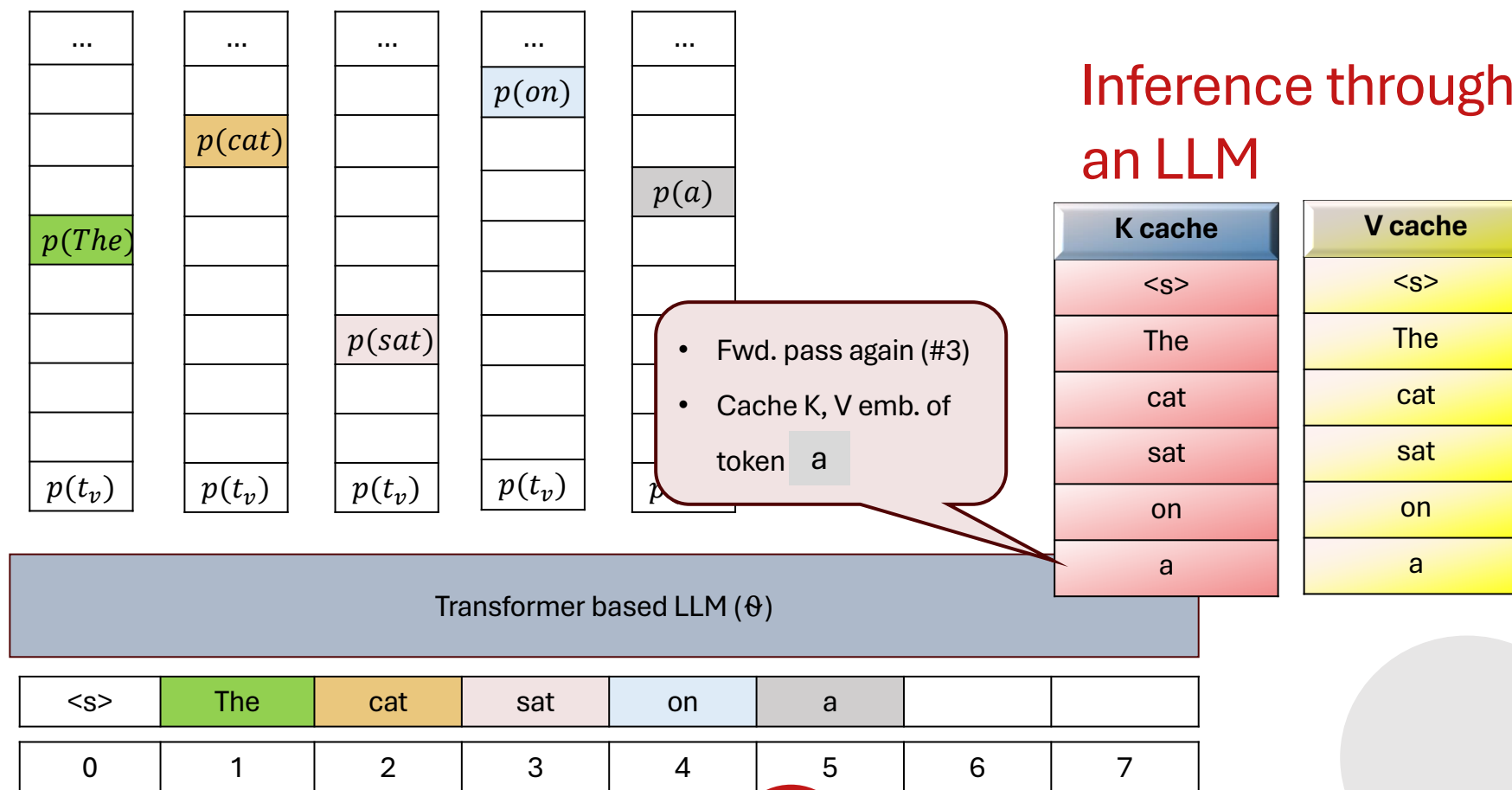
K cache	V cache
<s>	<s>
The	The
cat	cat
sat	sat
on	on

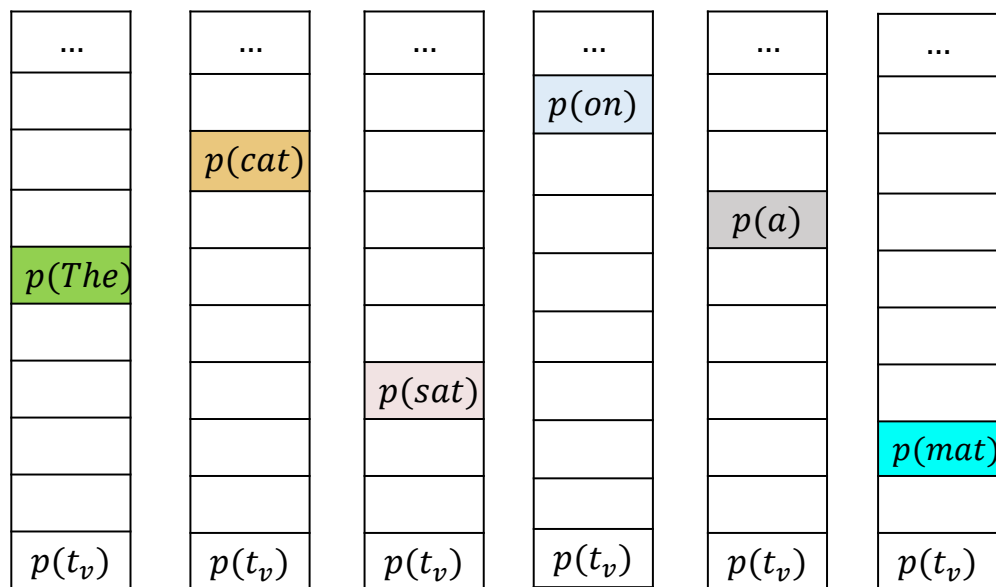
Transformer based LLM (θ)

<s>	The	cat	sat	on	a		
0	1	2	3	4	5	6	7

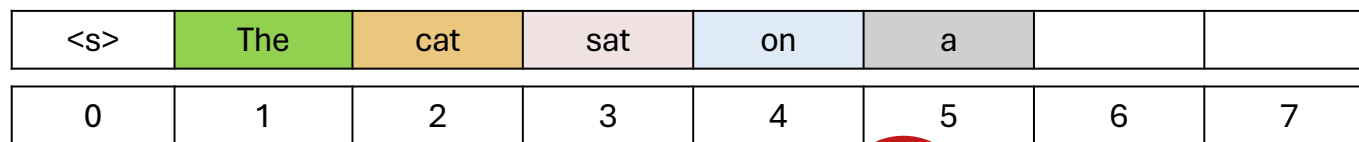
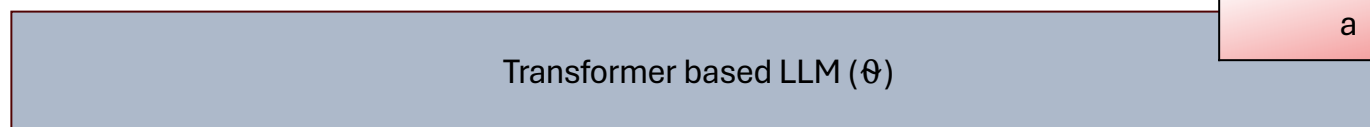
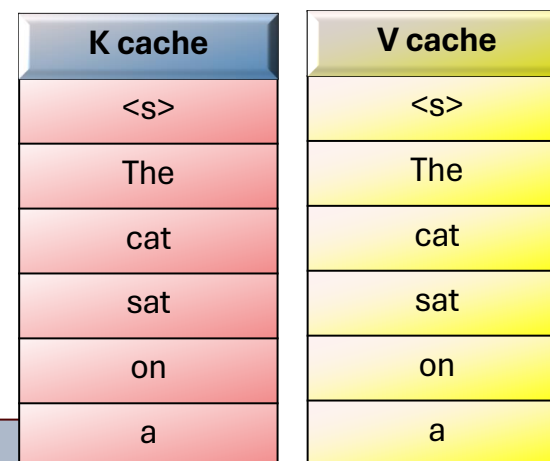


Inference through an LLM

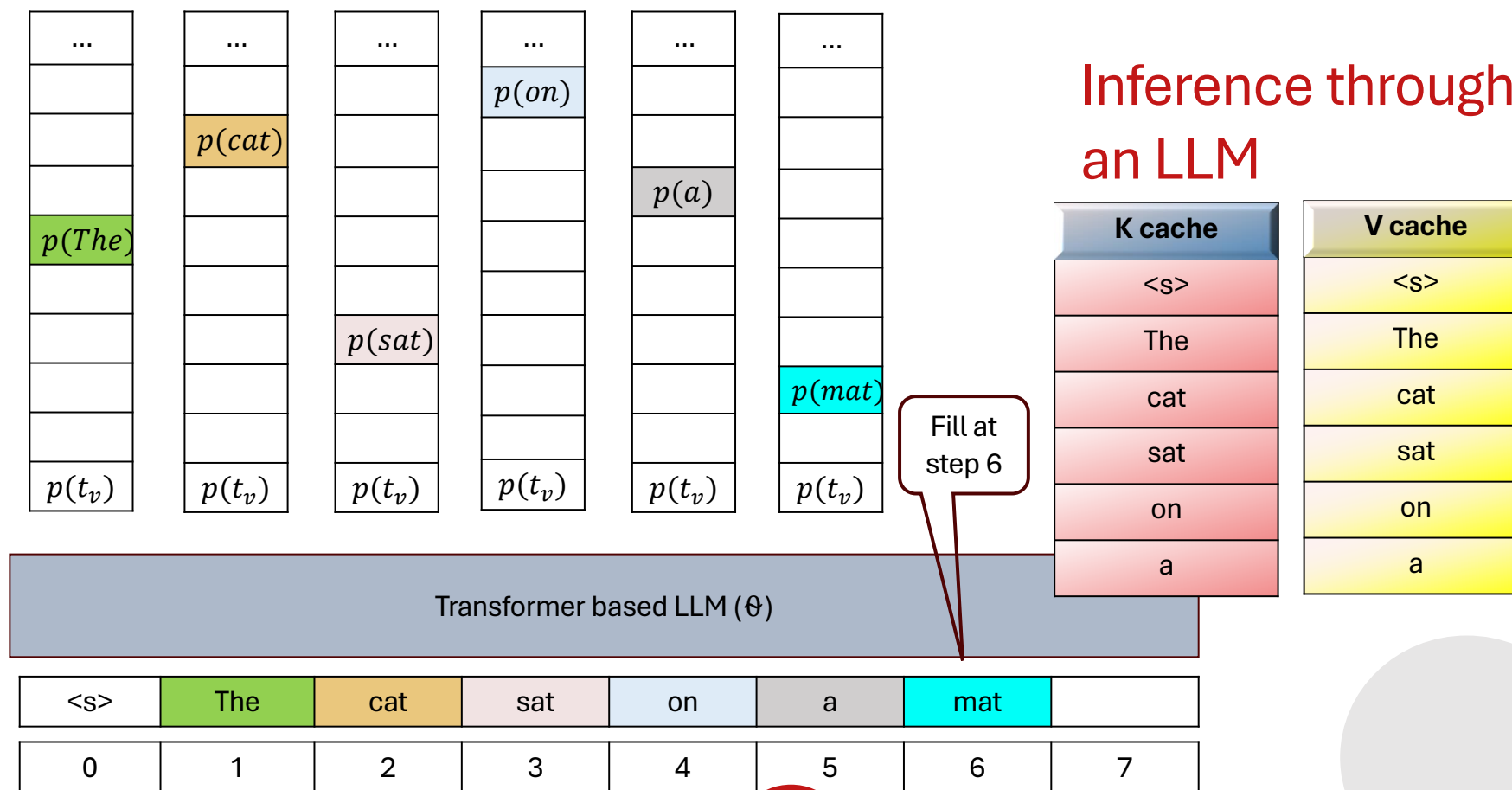




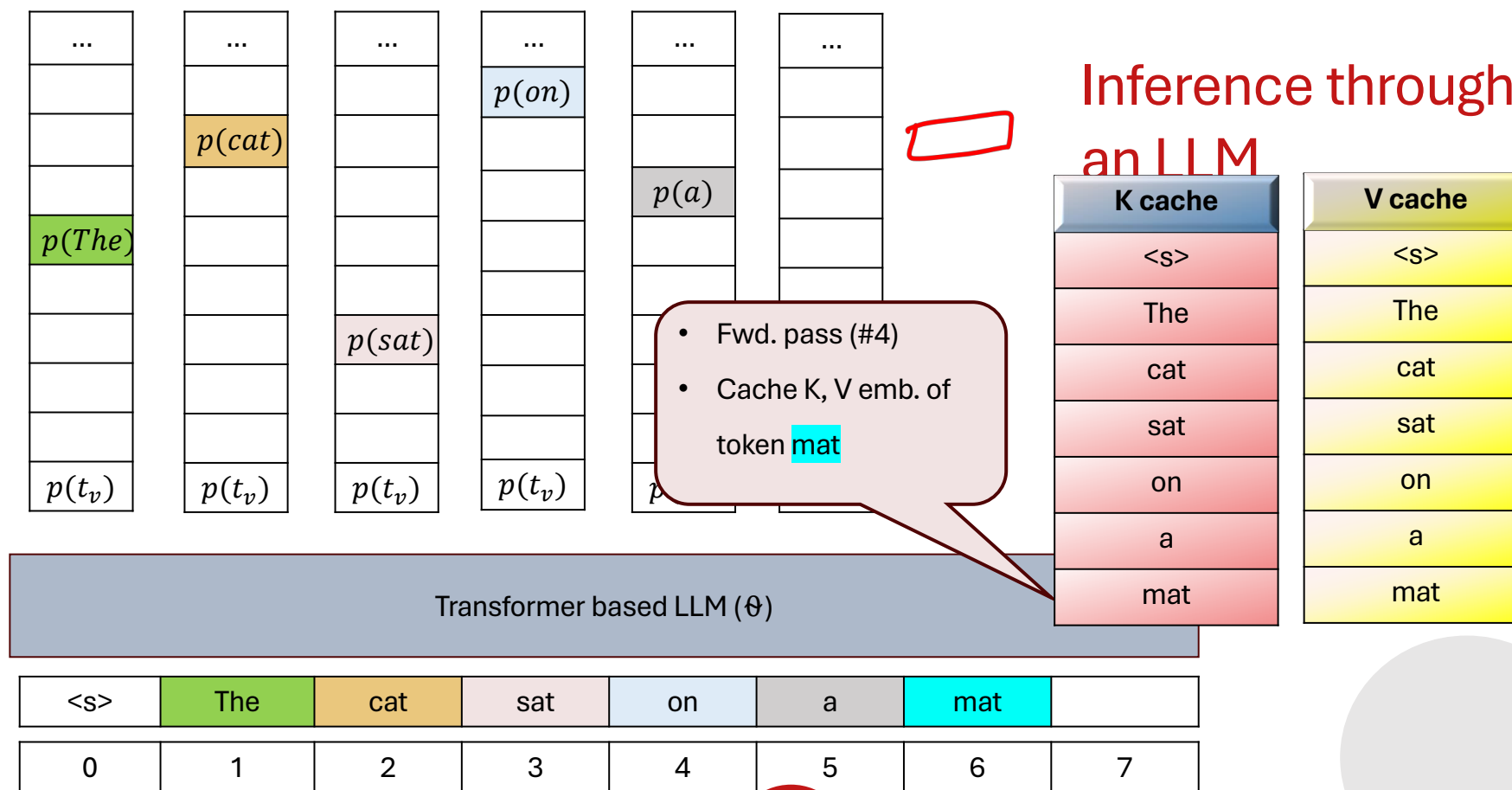
Inference through an LLM



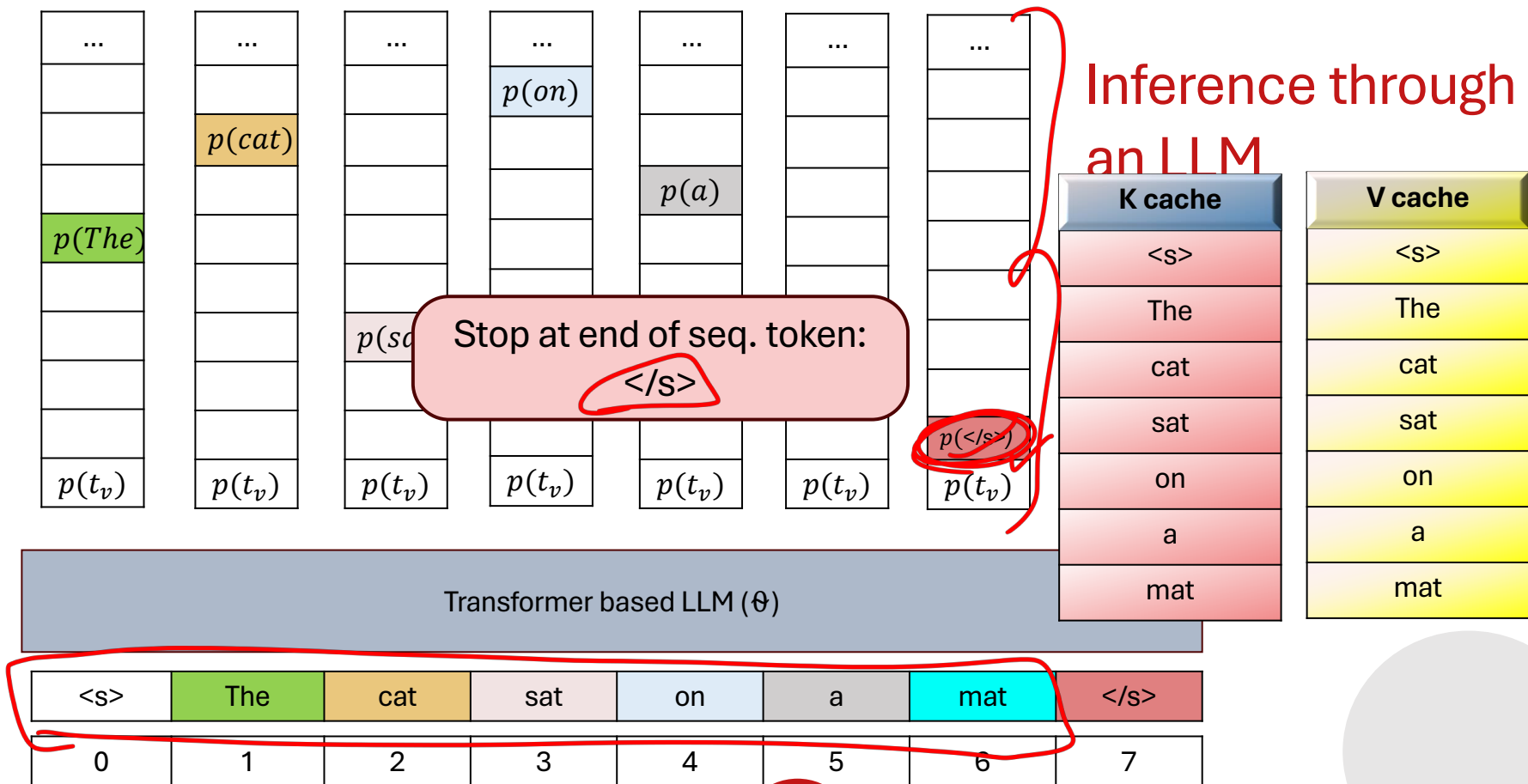
Inference through an LLM



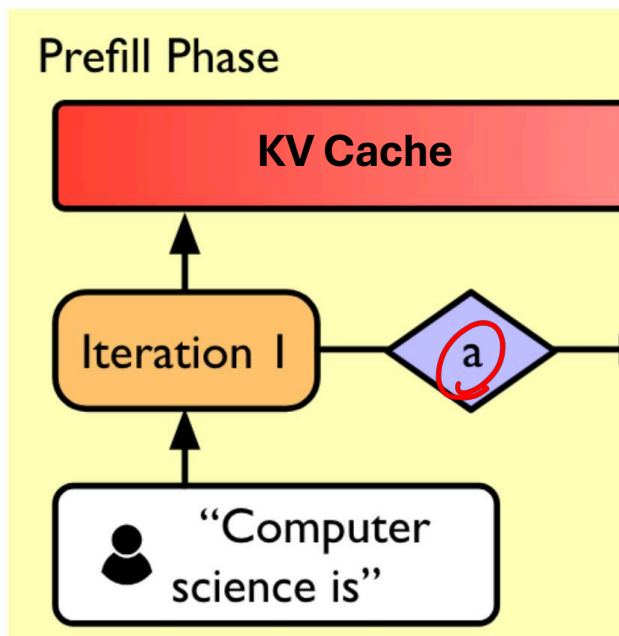
Inference through an LLM



Inference through an LLM



Two stages of LLM inference



- 1st forward pass (**Pre-fill step**) **Highly parallel**
 - ❖ The entire prompt is embedded and encoded – High latency
 - ❖ Multi-head attention computes the keys and values (KV)
 - ❖ Large matrix multiplication, high usage of the hardware accelerator

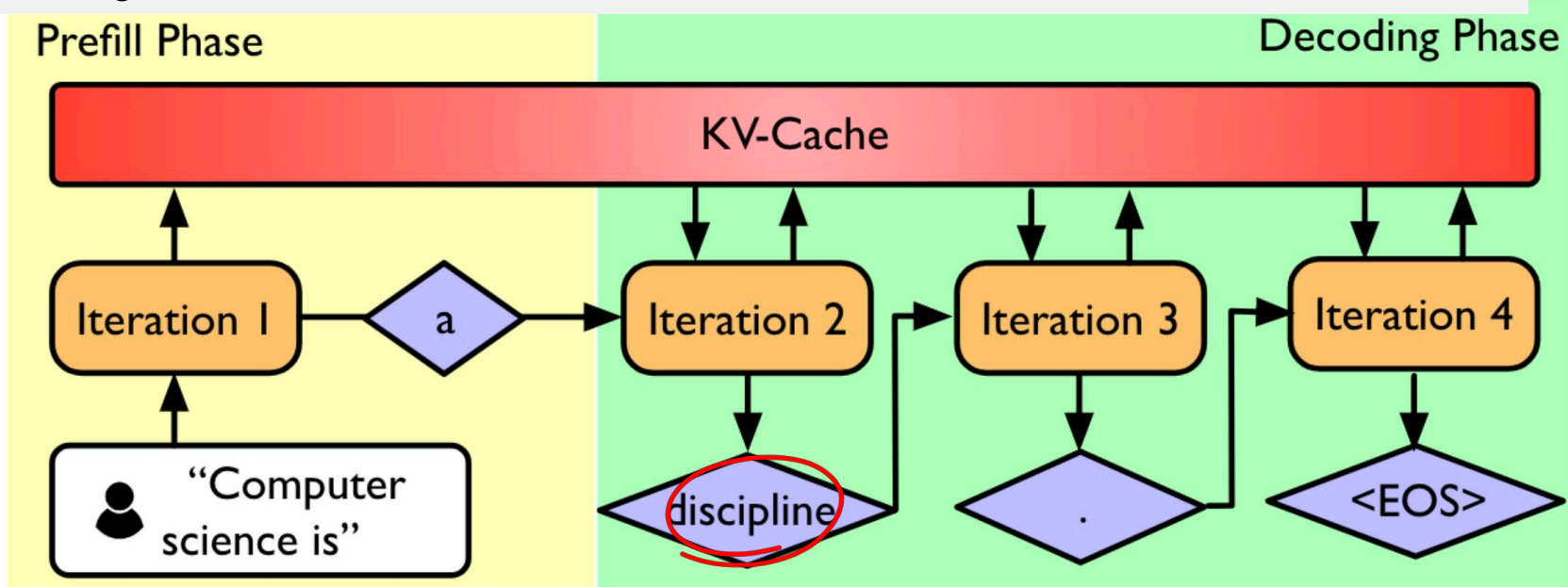


Content credits: Li et al, 2024 LLM Inference Serving: Survey of Recent Advances



Remaining forward passes (Output generation): **sequential**

- The answer is generated **one token** at a time – Low latency per step
- Each generated token is **appended** to the previous input
- The process is repeated until the **stopping criteria** is met (max. length or EOS)
- Low usage of the hardware accelerator



Content credits: Li et al, 2024 LLM Inference Serving: Survey of Recent Advances and Opportunities



Inference through an LLM

- 1st forward pass (**Pre-fill step**) **Highly parallel**
 - The entire prompt is embedded and encoded – High latency
 - Multi-head attention computes the keys and values (KV)
 - Large matrix multiplication, high usage of the hardware accelerator
- Remaining forward passes (**Output generation**): **sequential**
 - The answer is generated **one token** at a time – Low latency per step
 - Each generated token is **appended** to the previous input
 - The process is repeated until the **stopping criteria** is met (max. length or EOS)
 - Low usage of the hardware accelerator

Content credits: <https://www.slideshare.net/slideshow/julien-simon-deep-dive-optimizing-llm-inference-69d3/270921961>



Memory Usage of KV cache

$$2 * \text{precision} * N_{\text{layers}} * d_{\text{model}} * \text{seqlen} * \text{batch}$$

2	:	Two matrices for K and V
<i>precision</i>	:	bytes per parameter (e.g. 4 for fp32)
N_{layers}	:	layers in the model
d_{model}	:	dimension of embeddings
<i>seqlen</i>	:	length of context in tokens
<i>batch</i>	:	batch size

Content credits: https://www.youtube.com/watch?v=80blUggRj4&t=1s&ab_channel=EfficientNLP



Memory Usage of KV cache: **Example OPT-13B**

$$2 * precision * N_{layers} * d_{model} * seqlen * batch$$

2	:	Two matrices for K and V
<i>precision</i>	:	bytes per parameter (e.g. 4 for fp32)
N_{layers}	:	layers in the model
d_{model}	:	dimension of embeddings
<i>seqlen</i>	:	length of context in tokens
<i>batch</i>	:	batch size

2 (KV)
2 bytes (fp16)
40 layers
5120 dim.
2048 tokens
10

Content credits: https://www.youtube.com/watch?v=80blUggRJf4&t=1s&ab_channel=EfficientNLP



~ 26 GB

Memory Usage of KV cache: Example OPT-13B

$$2 * precision * N_{layers} * d_{model} * seq_{len} * batch$$

KV Cache: 17 GB

Model Size: $2 * 13 = 26$ GB

On a 40GB A100

- 65% (26GB) used by model parameters
- ~30% (12 GB) available for KV cache
- Expected throughput ~ 8 batch size of 2048 tokens

2 (KV)
2 bytes (fp16)
40 layers
5120 dim.
2048 tokens
10

Content credits: https://www.youtube.com/watch?v=80blUggRj4&t=1s&ab_channel=EfficientNLP



Memory Management of KV Cache

Prompt A : *“The cat sat”*

Max Tokens: 2048

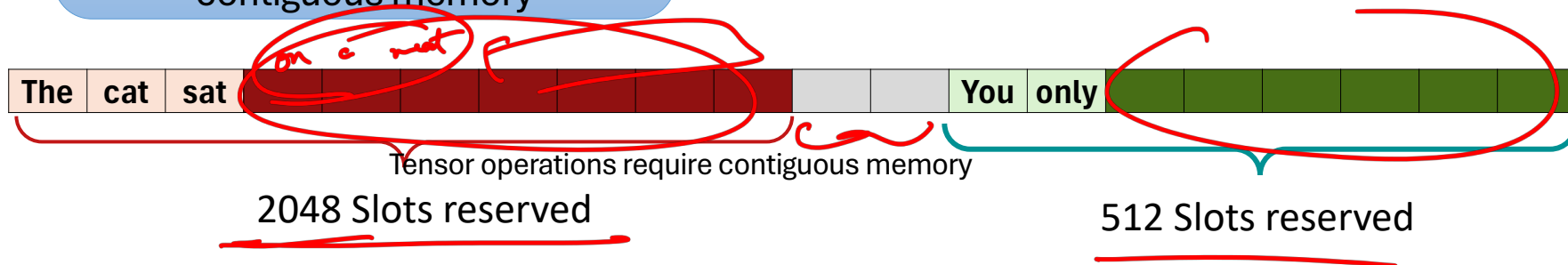
Prompt B : *“You only”*

Max Tokens: 512



Memory Management of KV Cache

Tensor operations require contiguous memory

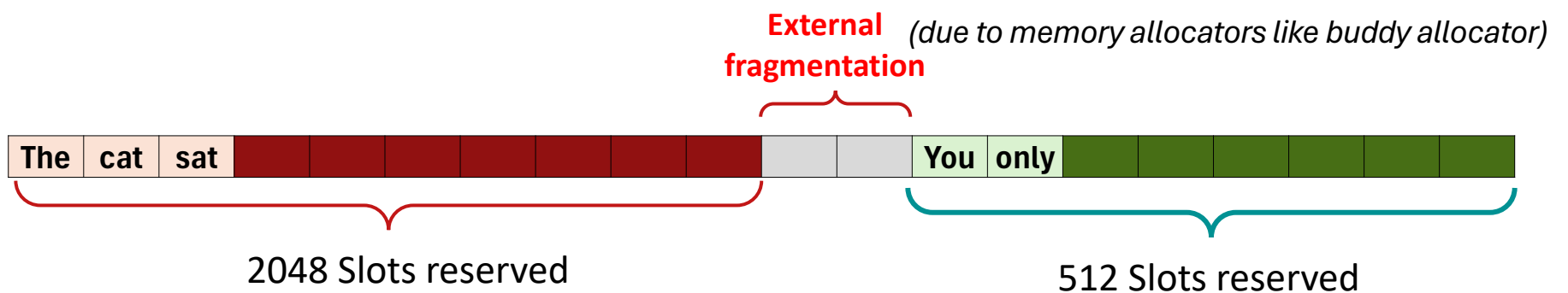


Prompt A: ***"The cat sat"***
Max Tokens: 2048

Prompt B: ***"You only"***
Max Tokens: 512



Memory Management of KV Cache



Prompt A : ***“The cat sat”***

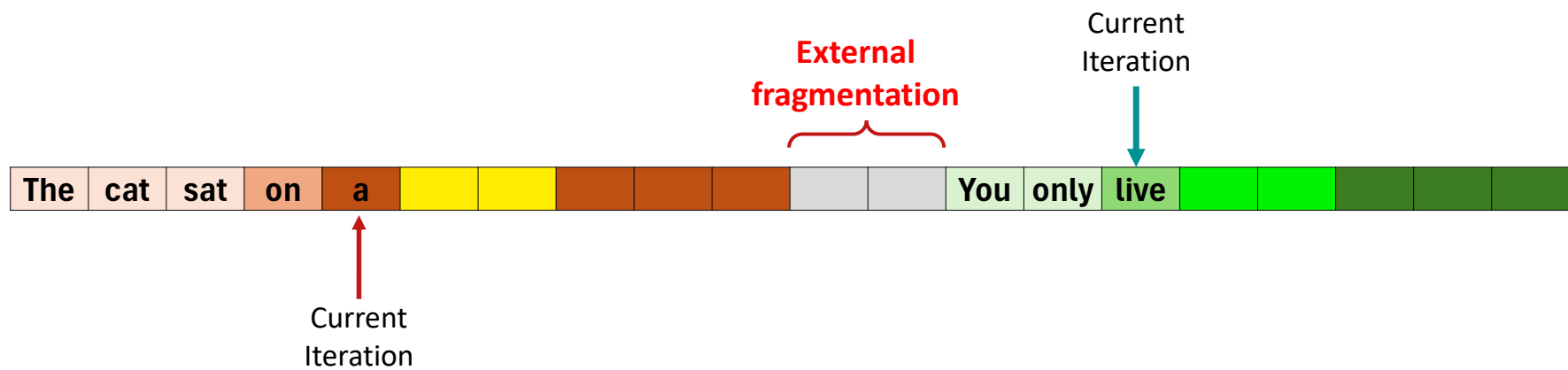
Max Tokens: ***2048***

Prompt B : ***“You only”***

Max Tokens: ***512***



Memory Management of KV Cache

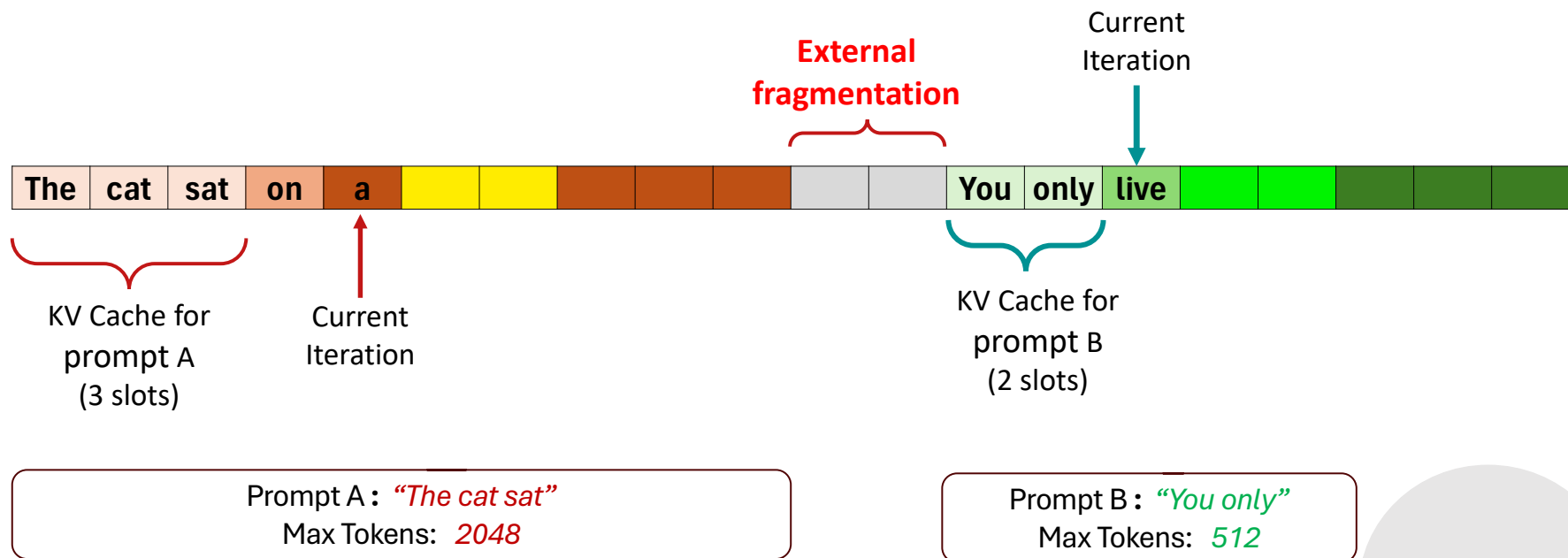


Prompt A : *"The cat sat"*
Max Tokens: 2048

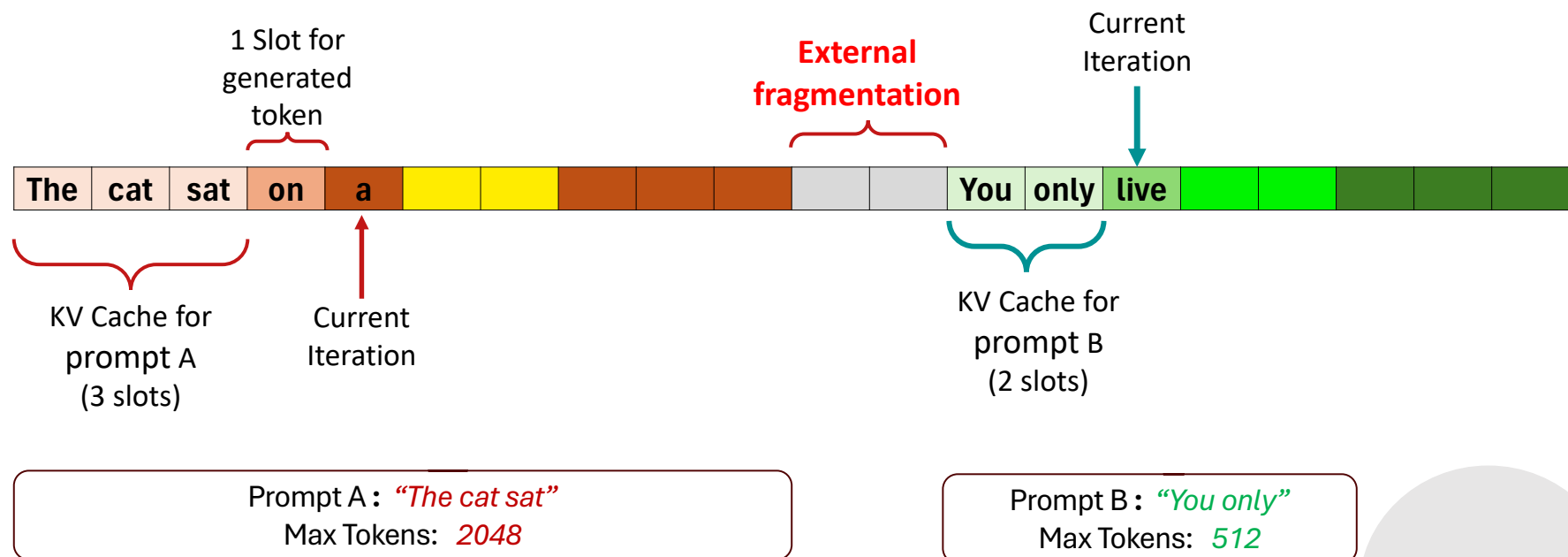
Prompt B : *"You only"*
Max Tokens: 512



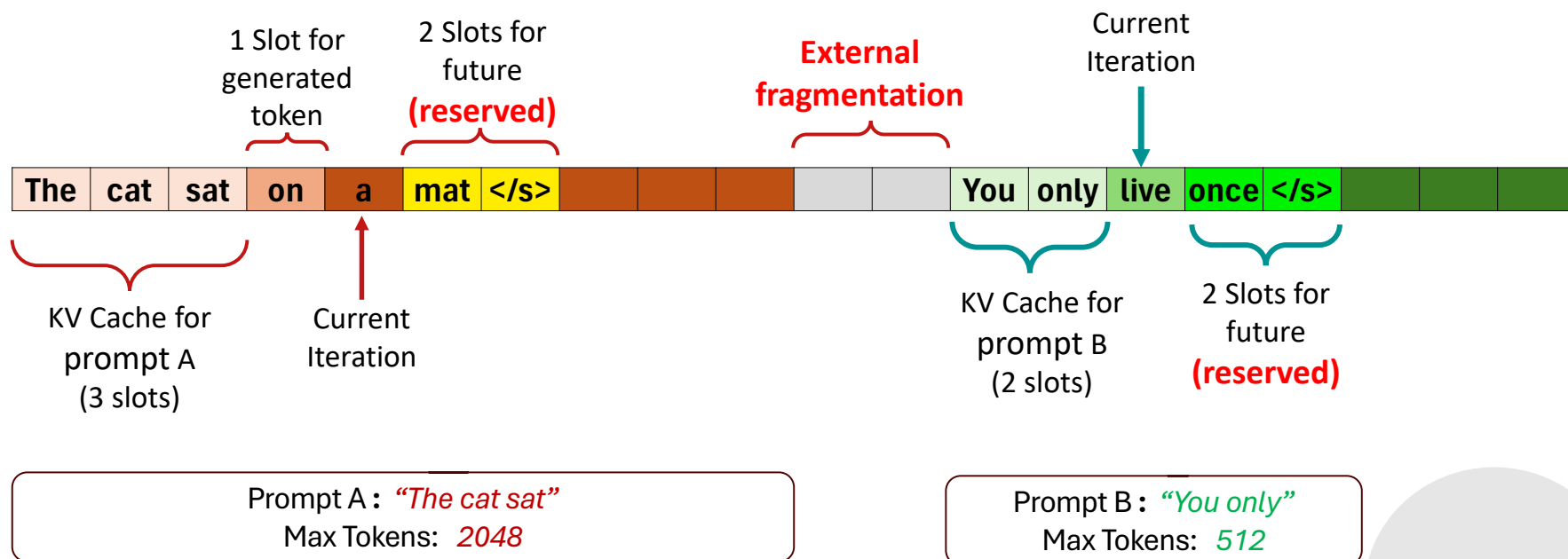
Memory Management of KV Cache



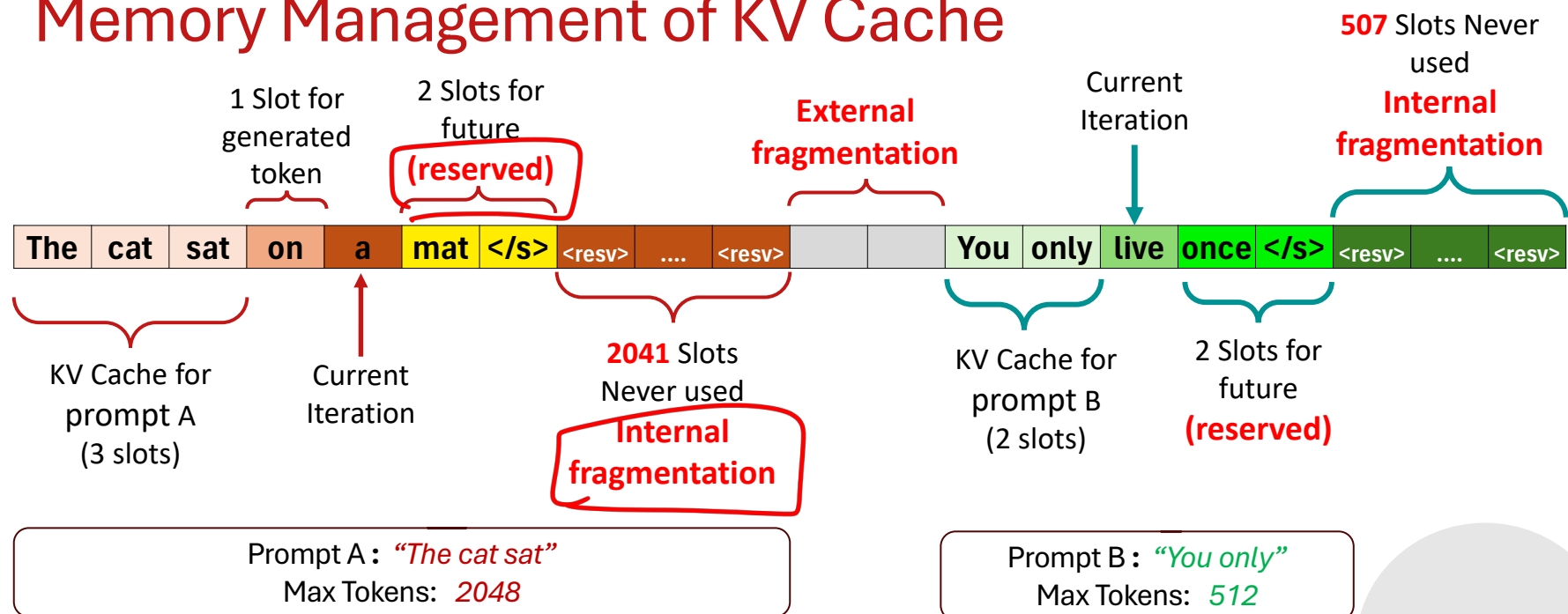
Memory Management of KV Cache



Memory Management of KV Cache



Memory Management of KV Cache



Memory Management of KV Cache

Chunk Pre-allocation scheme

- KV cache stored in contiguous memory
- Chunks of memory allocated statically, based on max. tokens.
- Actual input or eventual output length ignored while allocating memory



Memory Management of KV Cache

Chunk Pre-allocation scheme

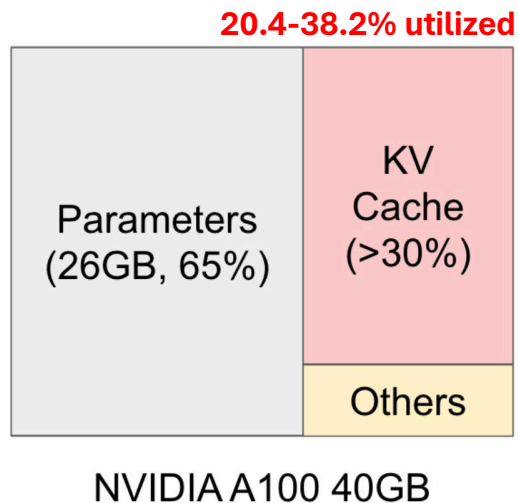
- KV cache stored in contiguous memory
- Chunks of memory allocated statically, based on max. tokens.
- Actual input or eventual output length ignored while allocating memory

Results in 3 types of memory wastes –

- **Reserved slots** for future tokens
- **Internal fragmentation** due to over-provisioning for maximum sequence lengths
- **External fragmentation** from the memory allocator.



Memory Layout for 13B-OPT model on A100 (40GB)

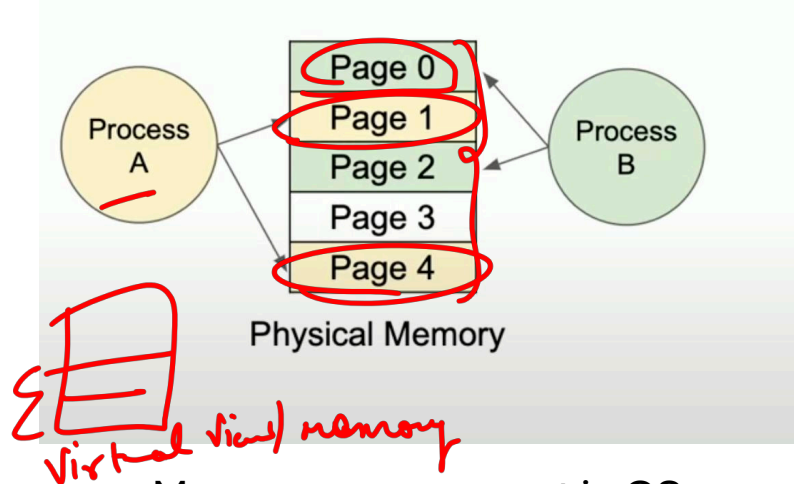


Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=Anyscale



vLLM: Efficient KV cache management

Inspired by **Virtual memory** and paging



Memory management in OS

Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=AnyScale



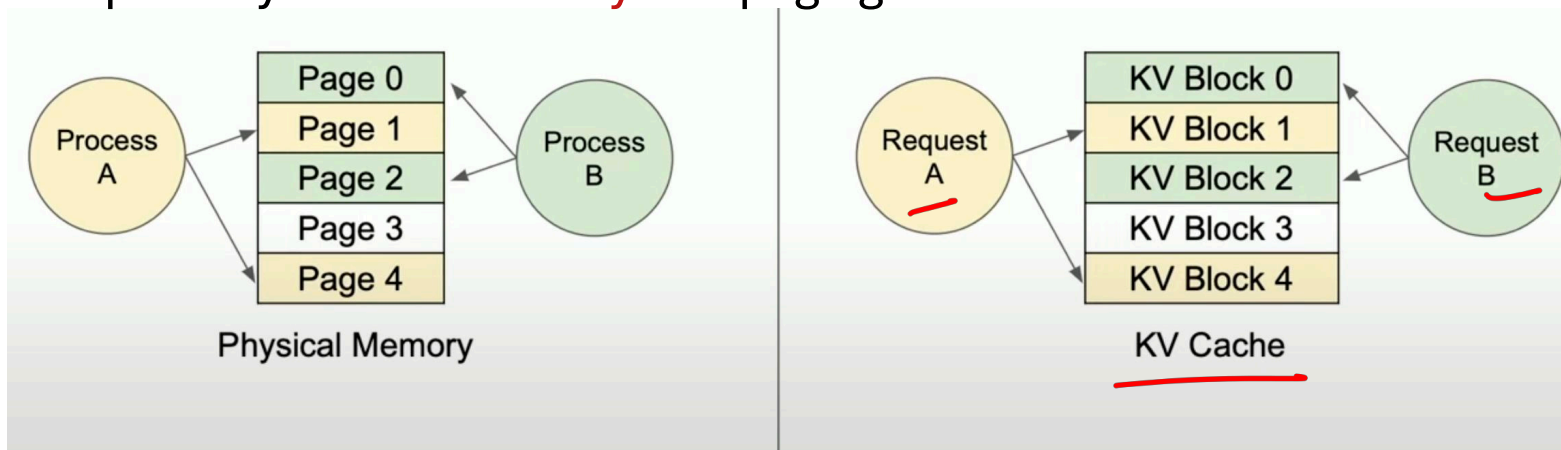
LLMs: Introduction and Recent Advances



Yatin Nandwani

vLLM: Efficient KV cache management

Inspired by **Virtual memory** and paging

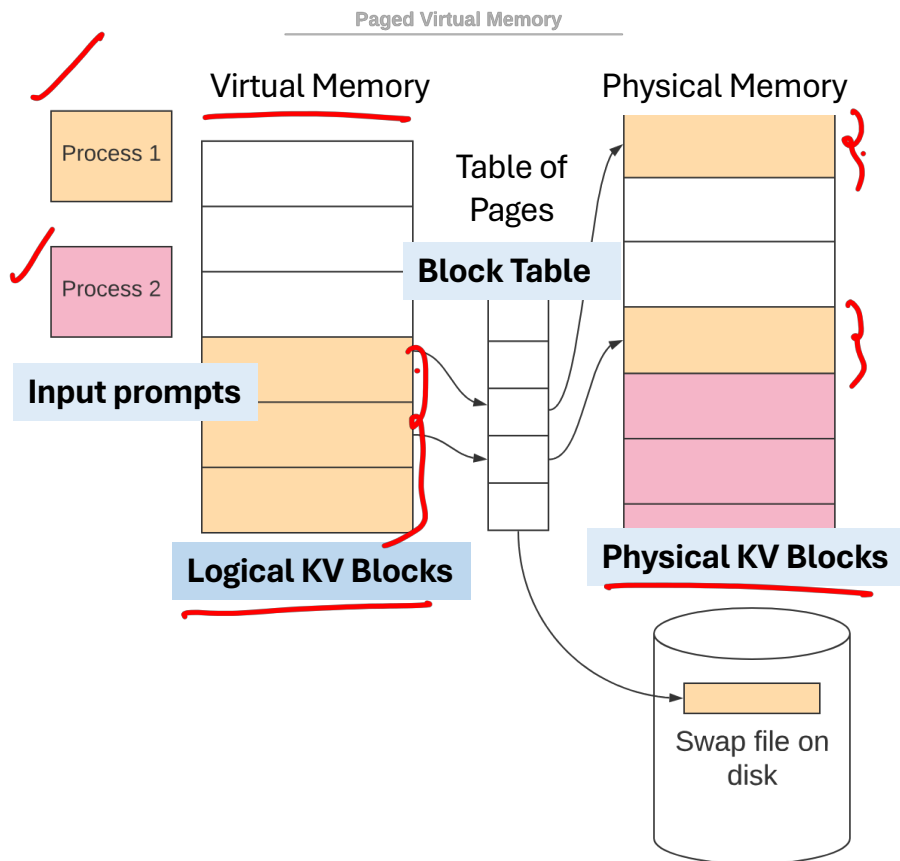


Memory management in OS

Memory management in vLLM

Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=AnyScale





Efficient KV cache management

Inspired by **Virtual memory** and paging

- ❑ Processes as **incoming requests** (input to the model)
- ❑ Virtual Memory to **Logical KV Blocks**
- ❑ Physical Memory to **Physical KV Blocks**
- ❑ Page table to **Block Table**

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



KV Blocks

KV Cache

Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=AnyScale

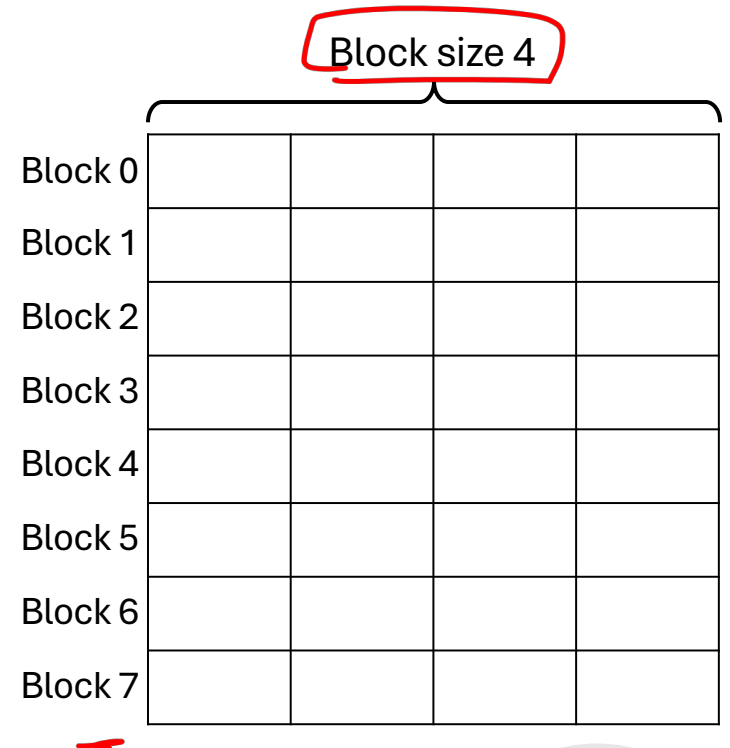


LLMs: Introduction and Recent Advances



Yatin Nandwani

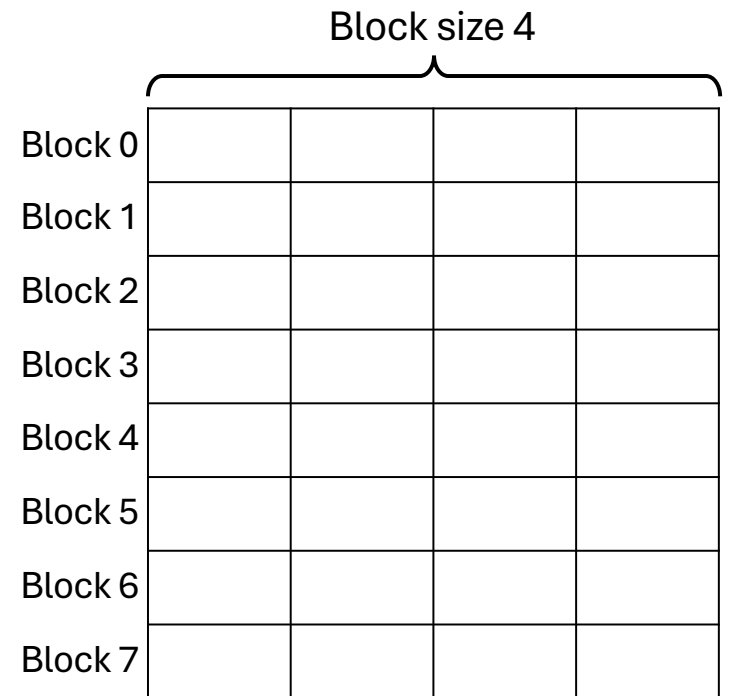
KV Blocks



Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=AnyScale



KV Blocks



Physical KV Blocks

Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=Anyscale



Physical vs Logical KV Blocks

Block 0				
1				
2				
3				

Logical KV Blocks

	Block size 4			
Block 0				
Block 1				
Block 2				
Block 3				
Block 4				
Block 5				
Block 6				
Block 7				

Physical KV Blocks

Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=AnyScale



Physical vs Logical KV Blocks

Block 0

1

2

3

Logical KV Blocks

Phys. Block	# Filled

Block Table

Block size 4

Block 0				
Block 1				
Block 2				
Block 3				
Block 4				
Block 5				
Block 6				
Block 7				

Physical KV Blocks

Content credits:

https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=Anyscale



LLMs: Introduction and Recent Advances



Yatin Nandwani

Physical vs Logical KV Blocks

Prompt: “Today we are learning about LLMs and”

Block 0	Today	we	are	learning
1	about	LLMs	and	
2				
3				

Logical KV Blocks

Phys. Block	# Filled

Block Table

	Block size 4			
Block 0				
Block 1				
Block 2				
Block 3				
Block 4				
Block 5				
Block 6				
Block 7				

Physical KV Blocks

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



Physical vs Logical KV Blocks

Prompt: “Today we are learning about LLMs and”

Block 0	Today	we	are	learning
1	about	LLMs	and	
2				
3				

Logical KV Blocks

Phys. Block	# Filled
7	4

Block Table

	Block size 4			
Block 0				
1				
2				
3				
4				
5				
6				
7	Today	we	are	learning

Physical KV Blocks

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



Physical vs Logical KV Blocks

Prompt: "Today we are learning about LLMs and"

Block 0

	Today	we	are	learning
1	about	LLMs	and	<u>memory</u>
2				
3				

Logical KV Blocks

Phys. Block	# Filled
7	4
1	3
	4

Block Table

Block size 4

Block 0

1	about	LLMs	and
2			
3			
4			
5			
6			
7	Today	we	are

Physical KV Blocks

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



Physical vs Logical KV Blocks

Prompt: “Today we are learning about LLMs and”

Completion: “*memory*”

Block 0	Today	we	are	learning
1	about	LLMs	and	memory
2				
3				

Logical KV Blocks

Phys. Block	# Filled
7	4
1	4

Block Table

	Block size 4			
Block 0				
1	about	LLMs	and	
2				
3				
4				
5				
6				
7	Today	we	are	learning

Physical KV Blocks

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



LLMs: Introduction and Recent Advances



Yatin Nandwani

Physical vs Logical KV Blocks

Prompt: “Today we are learning about LLMs and”

Completion: “*memory*”

Block 0	Today	we	are	learning
1	about	LLMs	and	memory
2				
3				

Logical KV Blocks

Phys. Block	# Filled
7	4
1	4

Block Table

	Block size 4			
Block 0				
1	about	LLMs	and	memory
2				
3				
4				
5				
6				
7	Today	we	are	learning

Physical KV Blocks

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



LLMs: Introduction and Recent Advances



Yatin Nandwani

Physical vs Logical KV Blocks

Prompt: "Today we are learning about LLMs and"

Completion: "memory on"

Block 0	Today	we	are	learning
1	about	LLMs	and	memory
2	on			
3				

Logical KV Blocks

Phys. Block	# Filled
7	4
1	4

Block Table

	Block size 4			
Block 0				
1	about	LLMs	and	memory
2				
3				
4				
5				
6				
7	Today	we	are	learning

Physical KV Blocks

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



LLMs: Introduction and Recent Advances

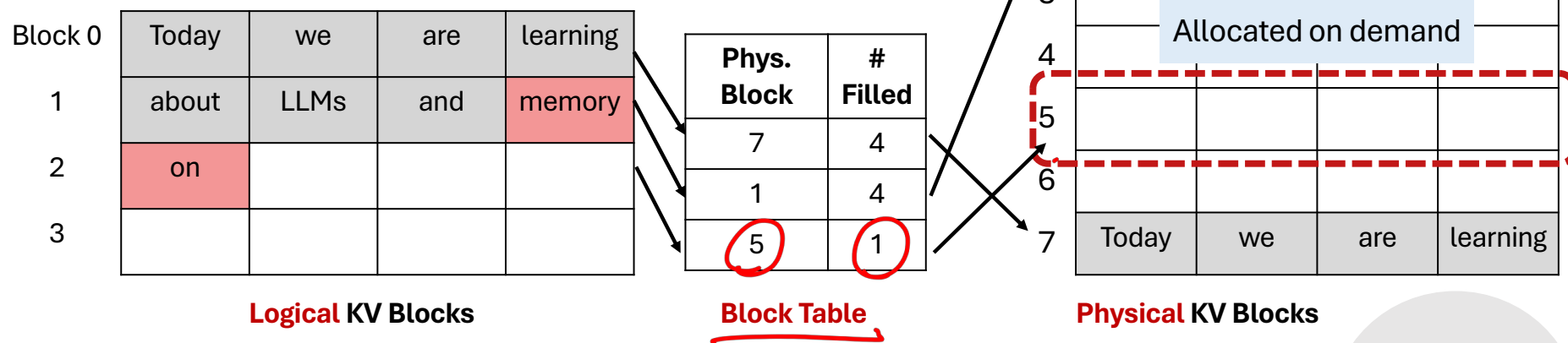


Yatin Nandwani

Physical vs Logical KV Blocks

Prompt: “Today we are learning about LLMs and”

Completion: “*memory on*”



Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



LLMs: Introduction and Recent Advances



Yatin Nandwani

Physical vs Logical KV Blocks

Prompt: “Today we are learning about LLMs and”

Completion: “*memory on*”

Block 0	Today	we	are	learning
1	about	LLMs	and	memory
2	on			
3				

Logical KV Blocks

Phys. Block	# Filled
7	4
1	4
5	1

Block Table

	Block size 4			
Block 0				
1	about	LLMs	and	memory
2				
3				
4				
5	on			
6				
7	Today	we	are	learning

Physical KV Blocks

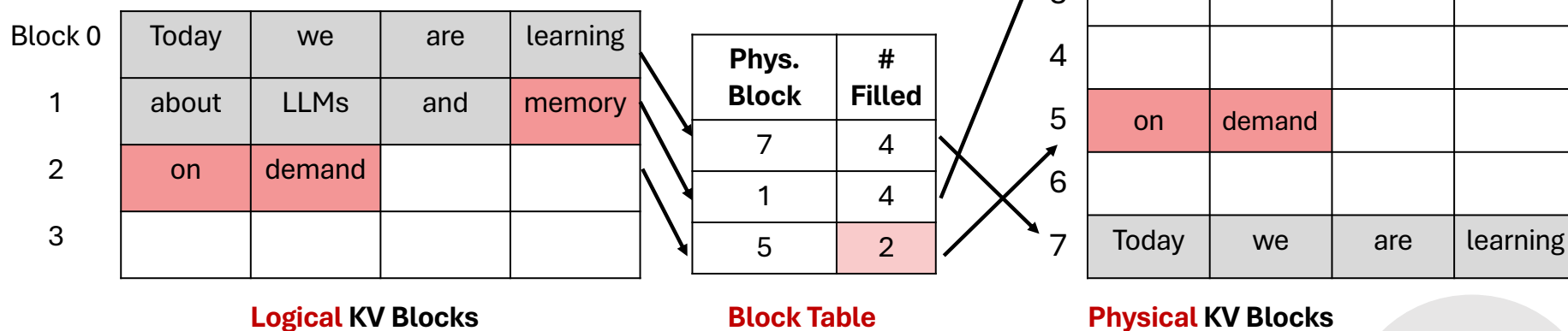
Allocated on demand



Physical vs Logical KV Blocks

Prompt: “Today we are learning about LLMs and”

Completion: “*memory on demand*”



Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>

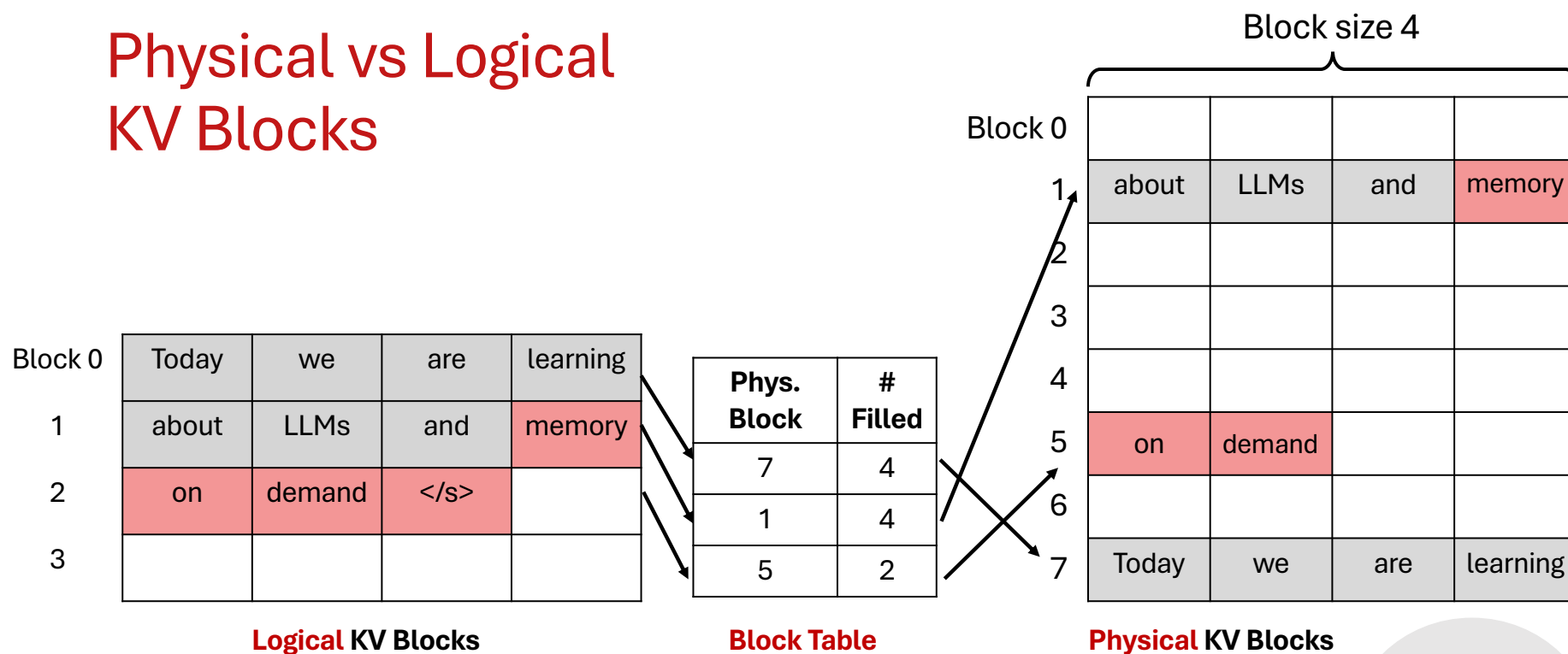


LLMs: Introduction and Recent Advances



Yatin Nandwani

Physical vs Logical KV Blocks



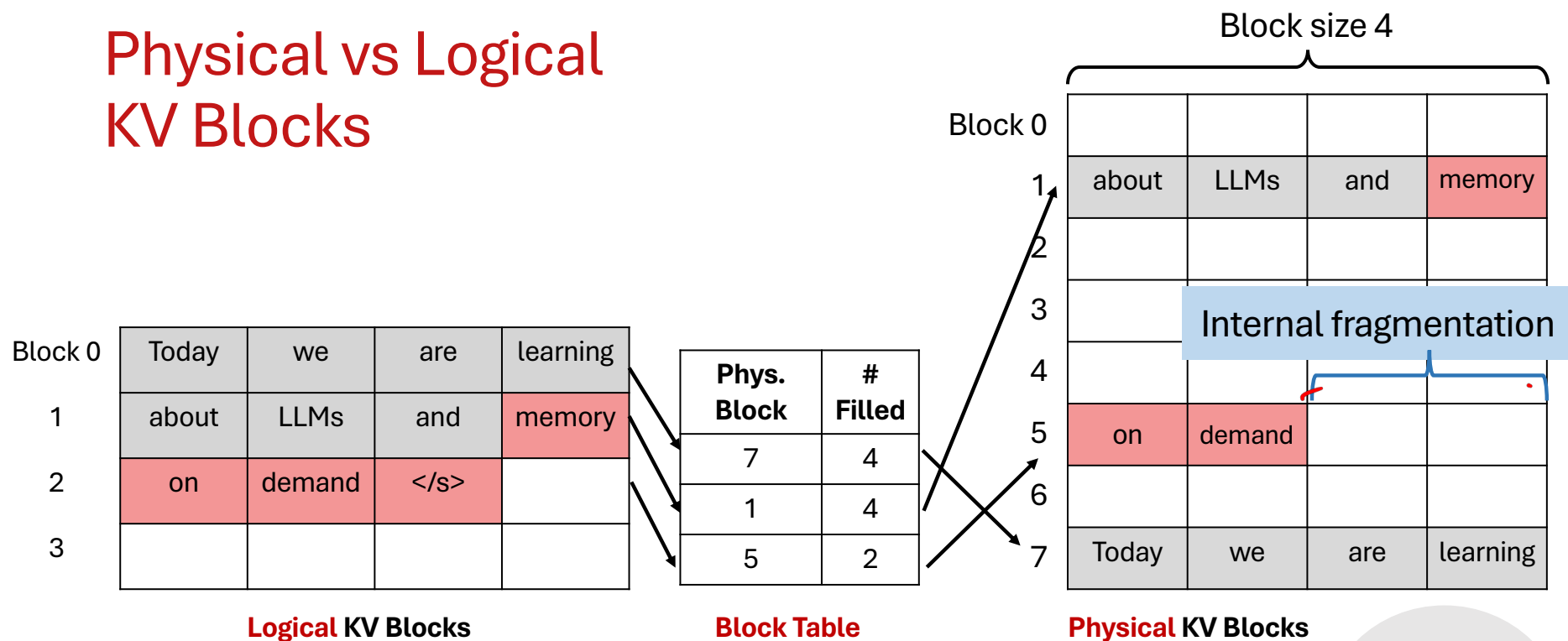
Prompt A: “Today we are learning about LLMs and”

Completion: “*memory on demand* </s>”

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



Physical vs Logical KV Blocks

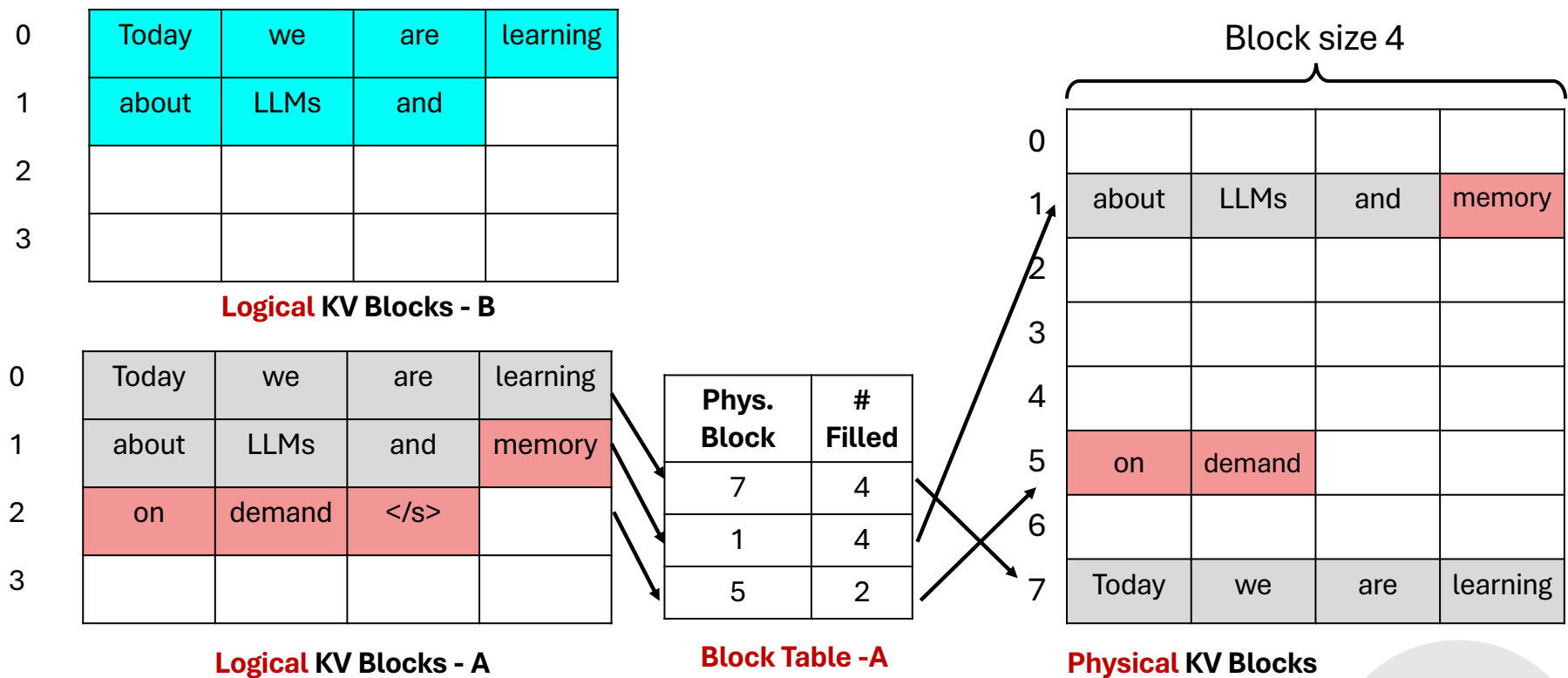


Prompt A: “Today we are learning about LLMs and”

Completion: “*memory on demand* </s>”

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>



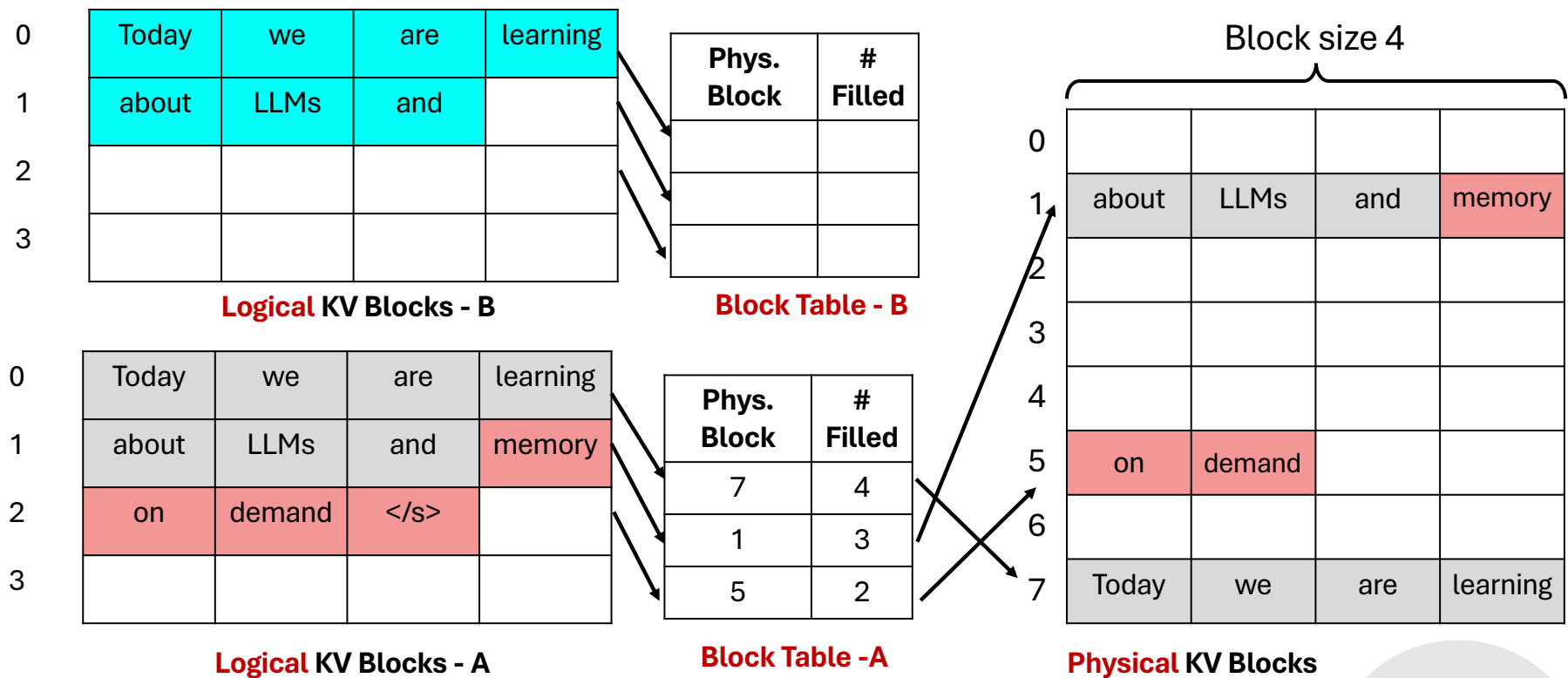


Prompt A: “Today we are learning about LLMs and”
Completion: “*memory on demand</s>*”

Prompt B: “Today we are learning about LLMs and”
Completion:

Content credits: <https://youtu.be/yVXtLTcdO1Q?si=XO2Dk-VYOShUMH1u>

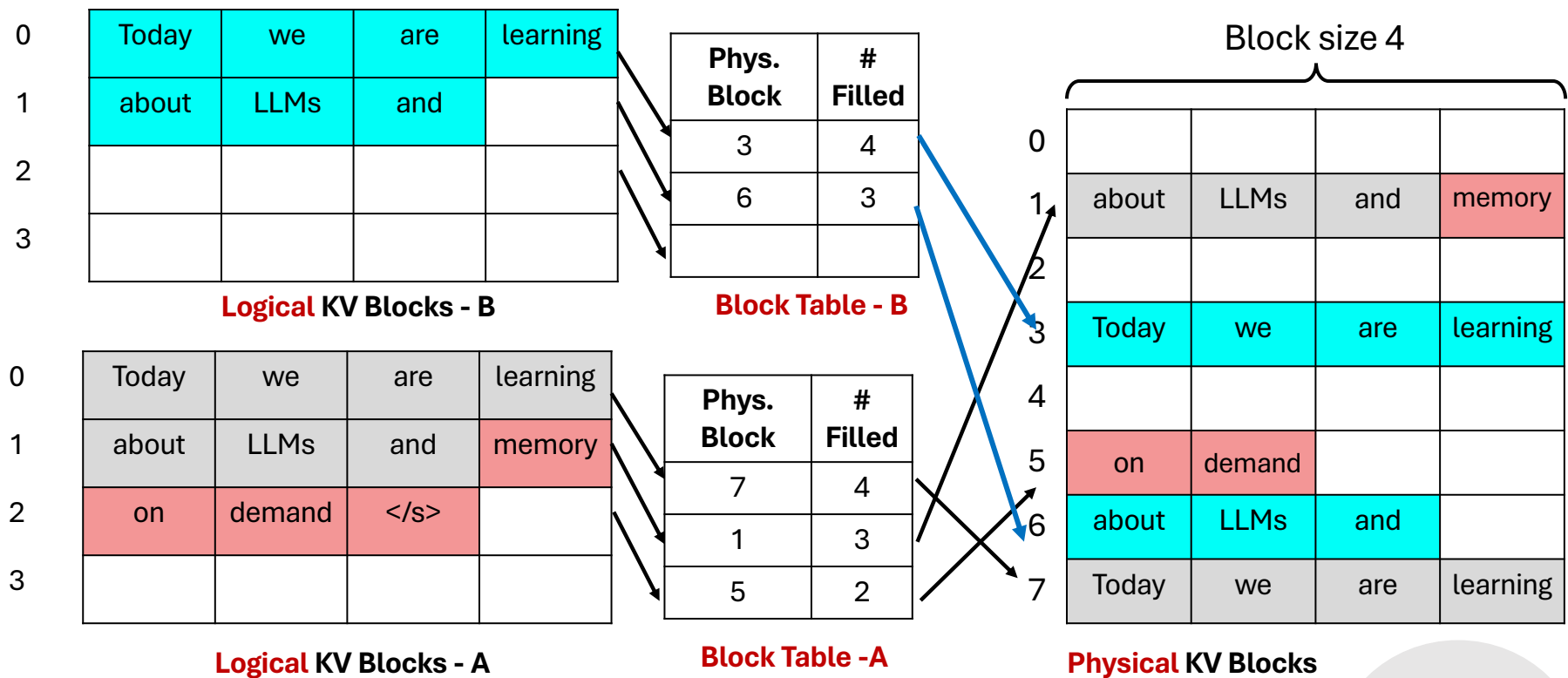




Prompt A: “Today we are learning about LLMs and”
Completion: “*memory on demand </s>*”

Prompt B: “Today we are learning about LLMs and”
Completion:

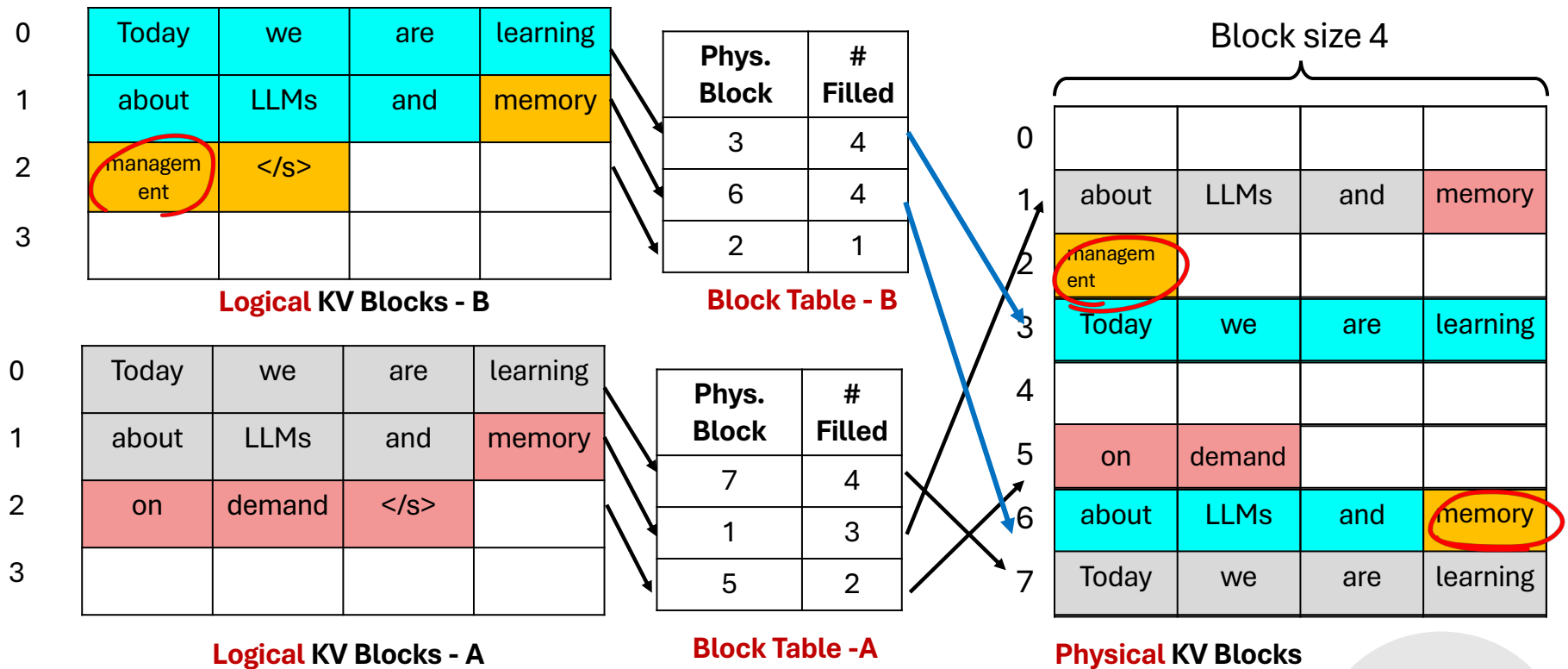




Prompt A: “Today we are learning about LLMs and”
Completion: “*memory on demand </s>*”

Prompt B: “Today we are learning about LLMs and”
Completion:



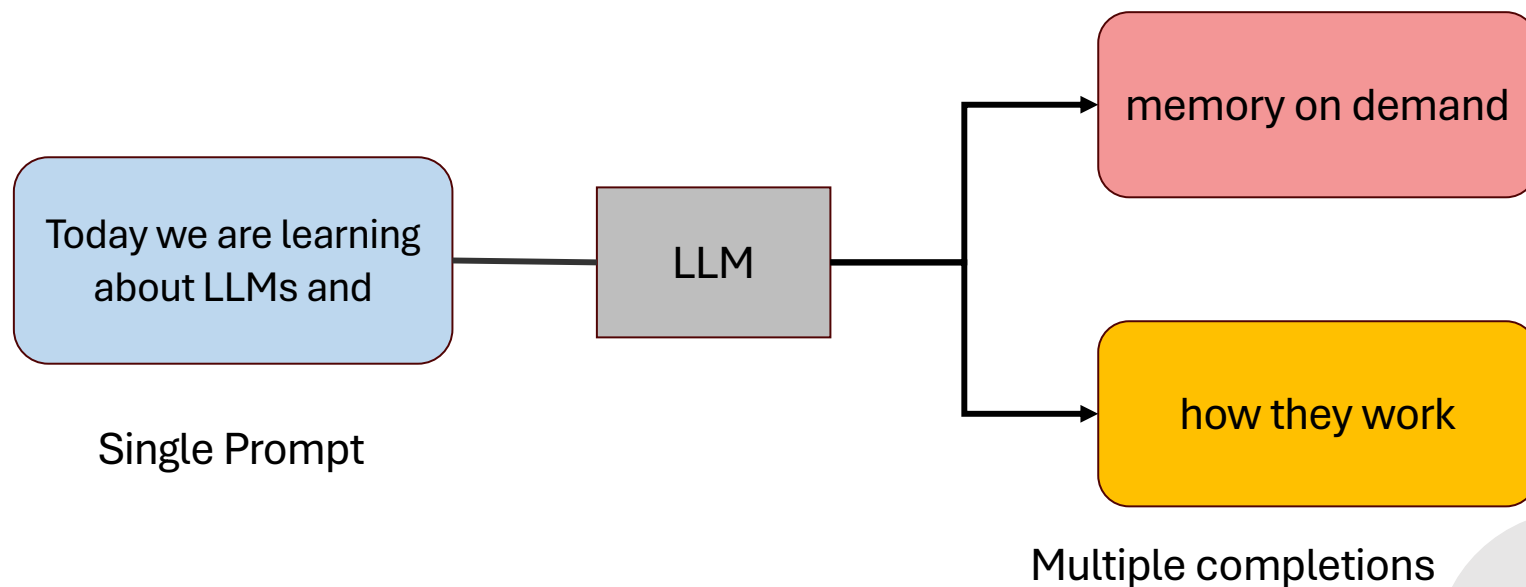


Prompt A: "Today we are learning about LLMs and"
Completion: "memory on demand </s>"

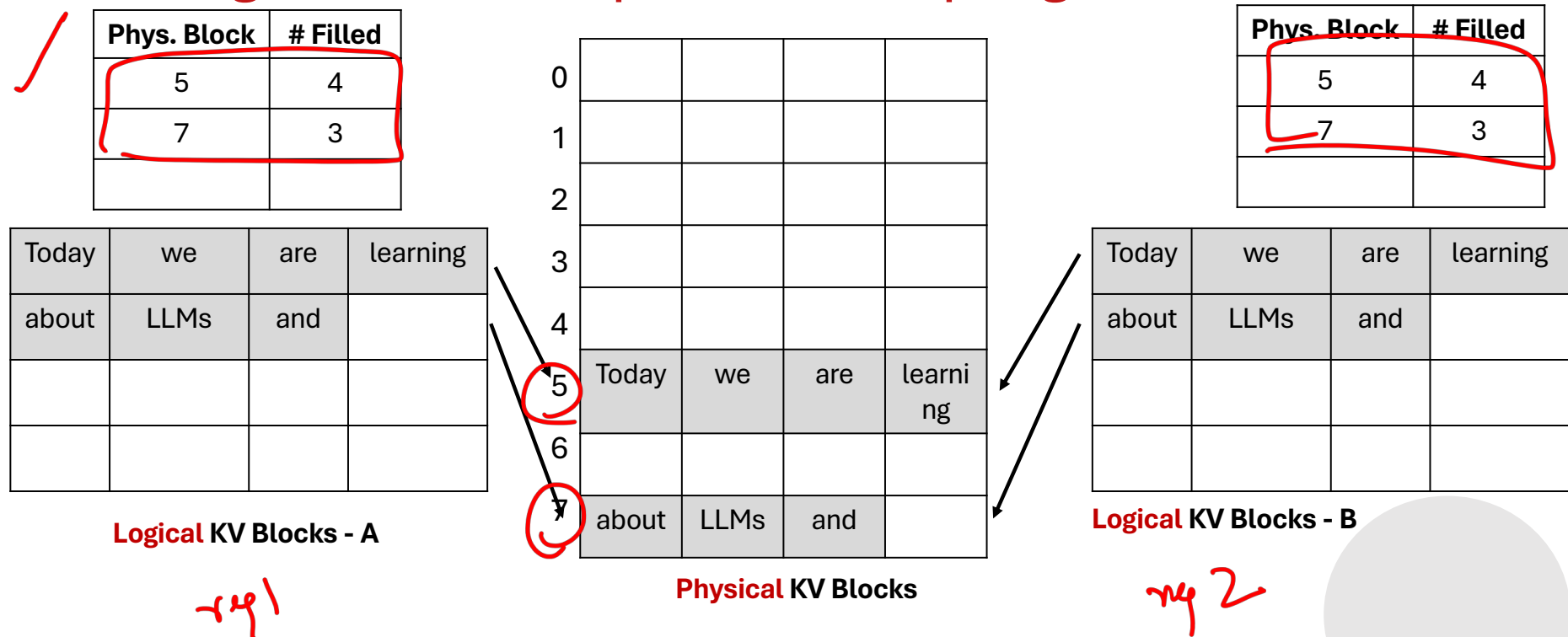
Prompt B: "Today we are learning about LLMs and"
Completion: "memory management </s>"



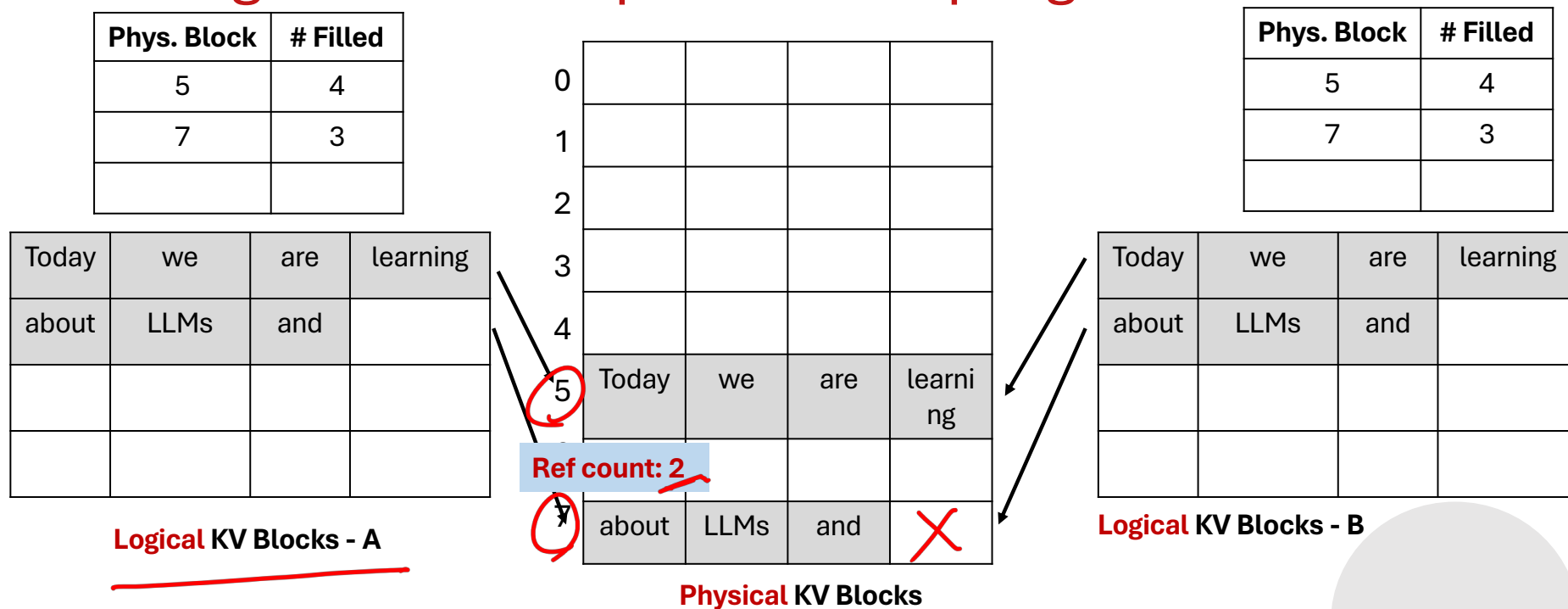
Dynamic block mapping enables sharing



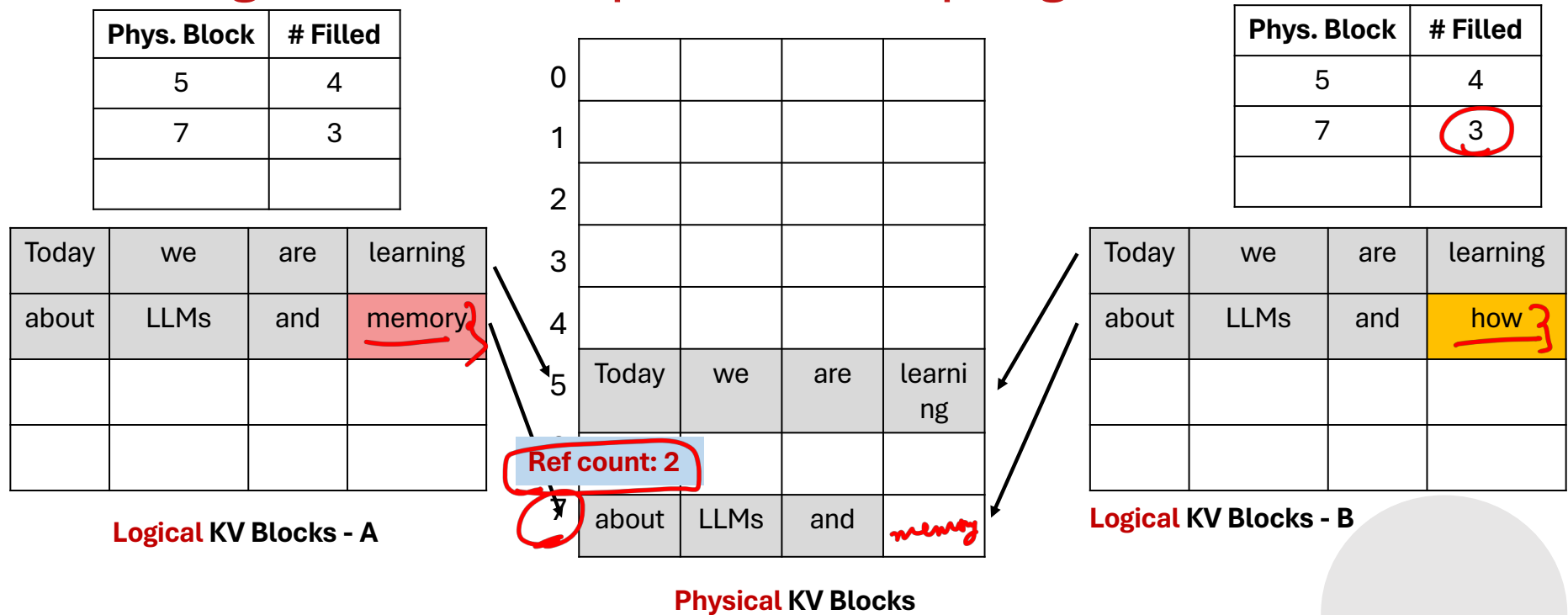
Sharing KV blocks in parallel sampling



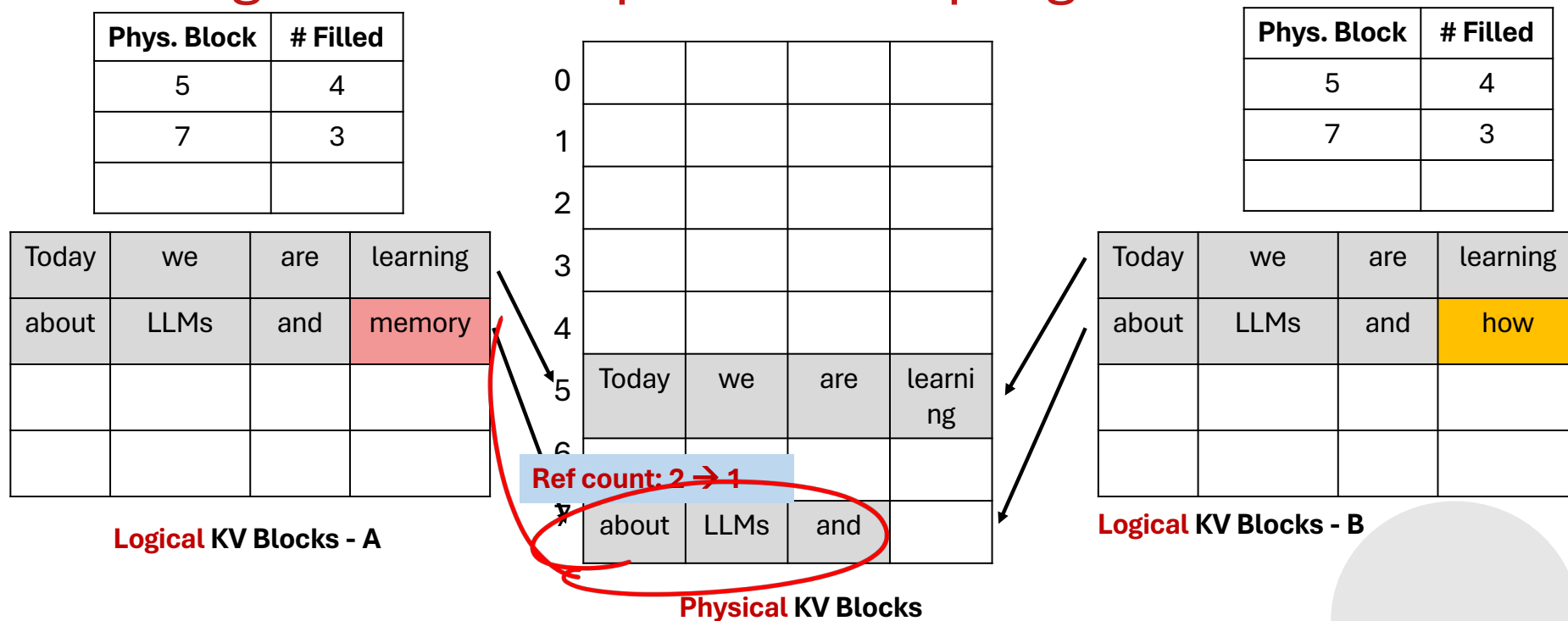
Sharing KV blocks in parallel sampling



Sharing KV blocks in parallel sampling

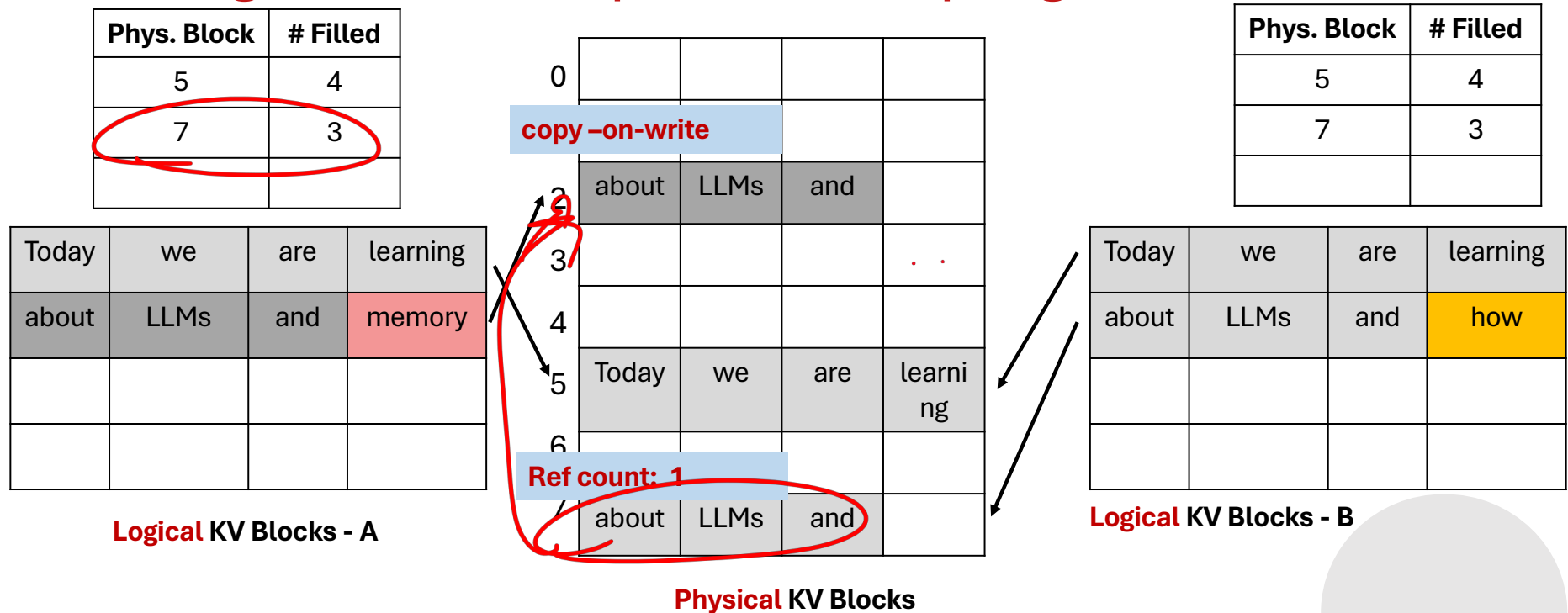


Sharing KV blocks in parallel sampling

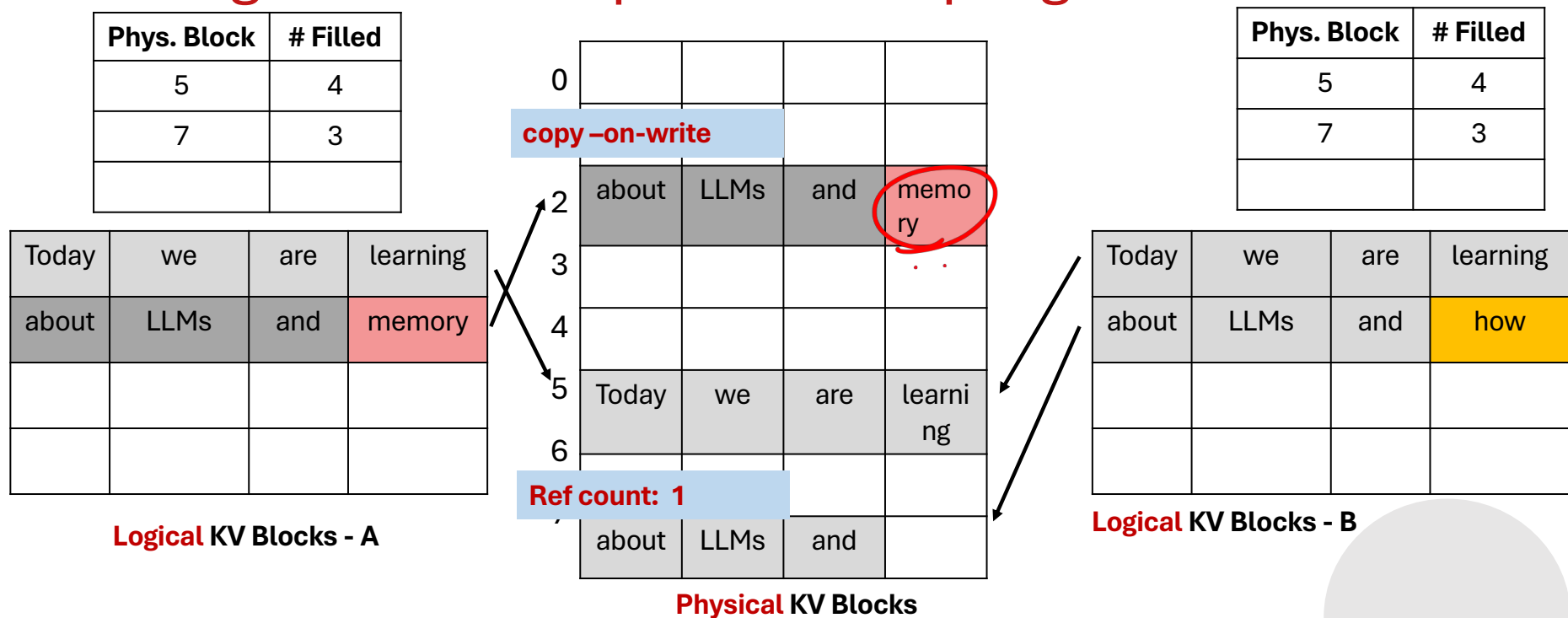


Copy-on-write

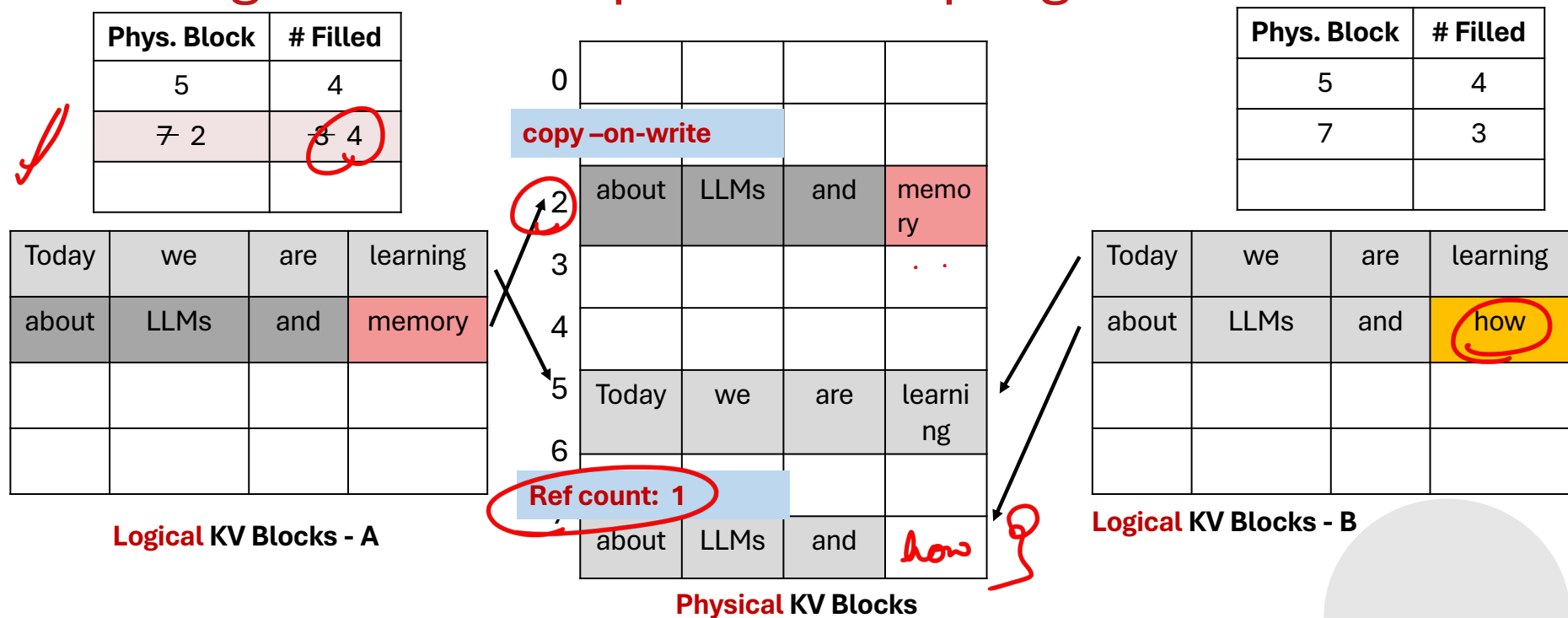
Sharing KV blocks in parallel sampling



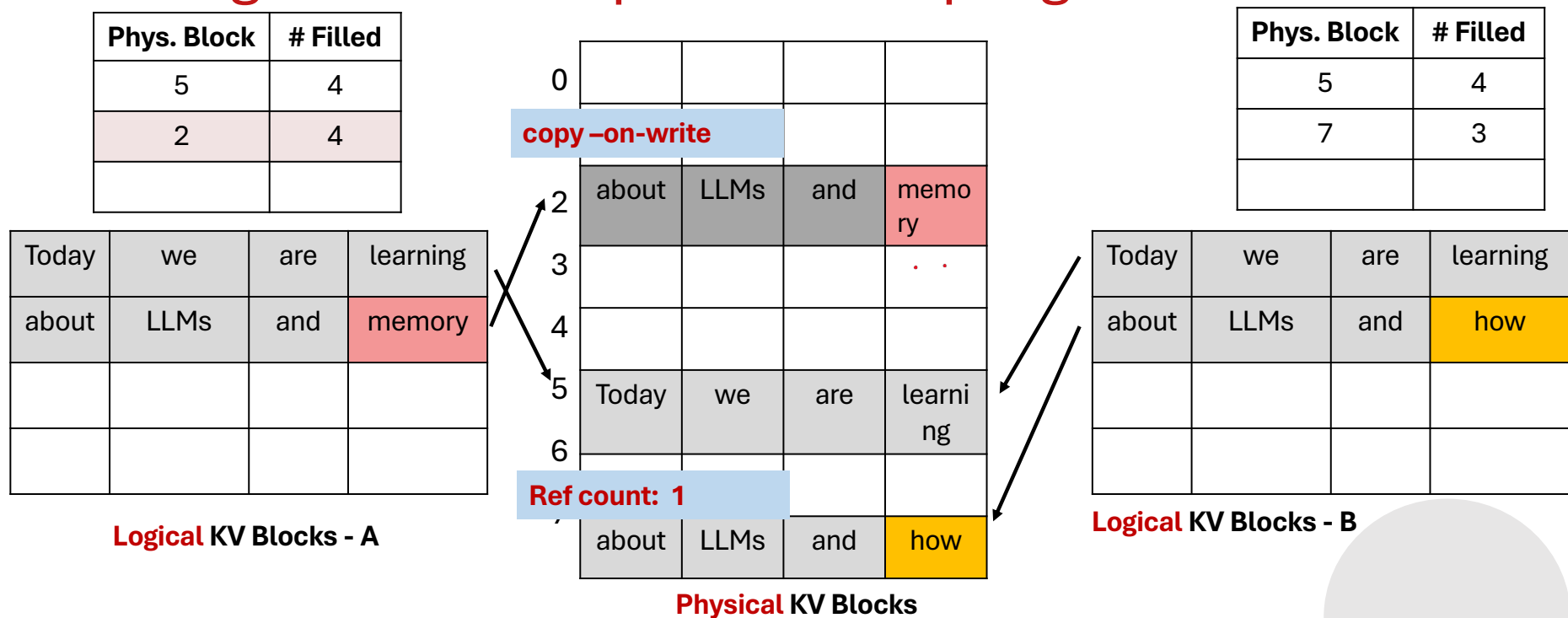
Sharing KV blocks in parallel sampling



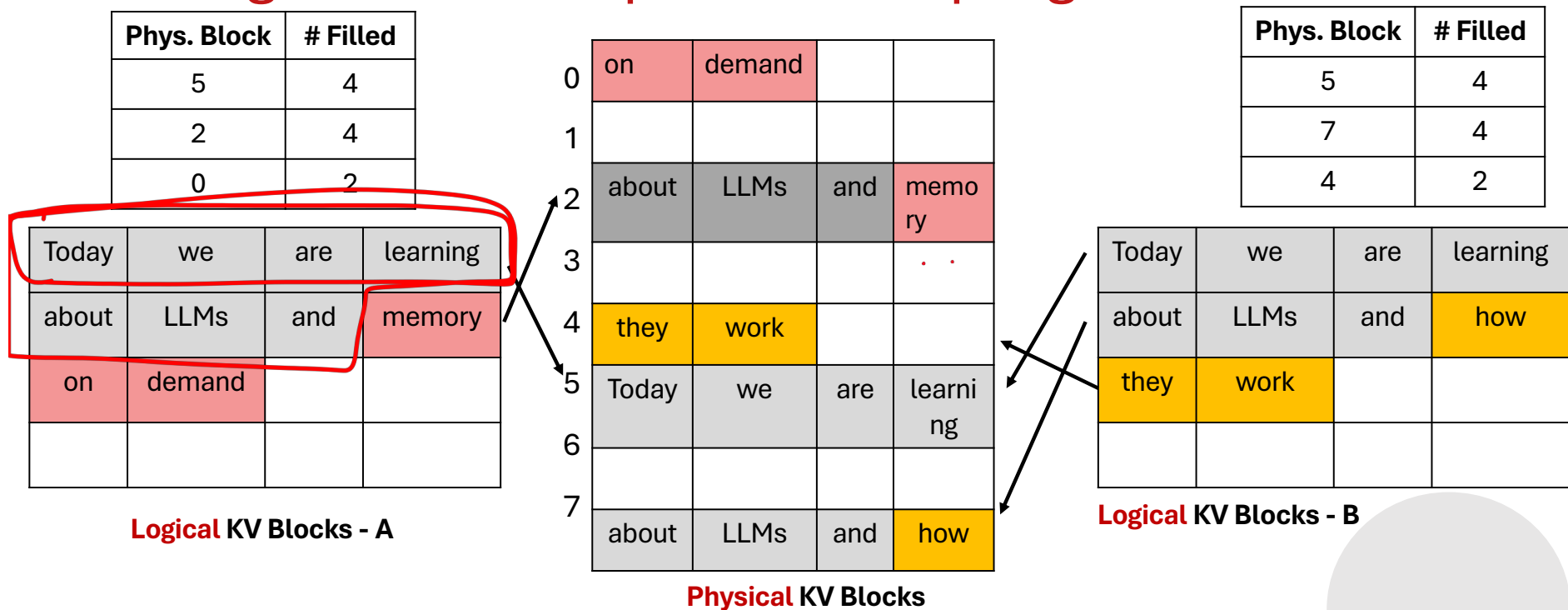
Sharing KV blocks in parallel sampling



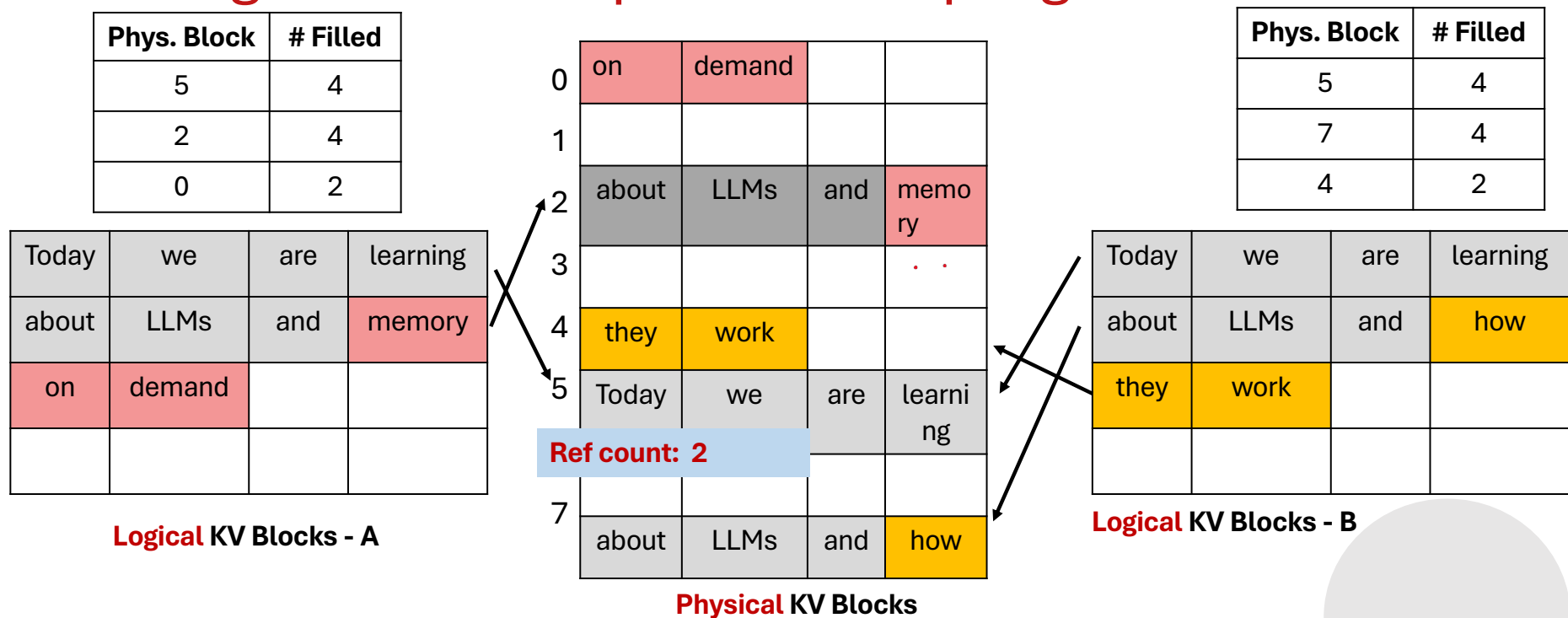
Sharing KV blocks in parallel sampling



Sharing KV blocks in parallel sampling



Sharing KV blocks in parallel sampling



Memory efficiency of vLLMs

✓ Minimal internal fragmentation

- Only happens at the last block of a sequence

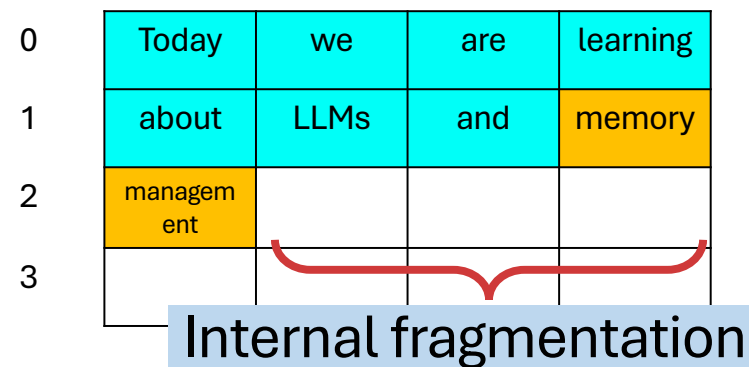
○ **# wasted tokens / seq < block size**

- Sequence: O(100) or O(1000) tokens
- Block size: 16 or 32 tokens

✓ No external fragmentation

✓ On average, wasted space < **4%** of KV cache

✓ **3-5x** improved memory utilization!



Content credits: https://www.youtube.com/watch?v=5ZlavKF_98U&t=1646s&ab_channel=AnyScale



softmax $\begin{bmatrix} a_1 & a_4 \end{bmatrix}$
 $\begin{bmatrix} p_1 & p_4 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \frac{\sum p_i v_i}{\sum e^{a_i}} = \frac{p_1 v_1 + p_2 v_2 + p_3 v_3 + p_4 v_4}{e^{a_1} + e^{a_2} + e^{a_3} + e^{a_4}}$

Paged Attention

- Tensor operations require contiguous memory
- How to compute attention softmax across fragmented memory?
- Paged Attention!

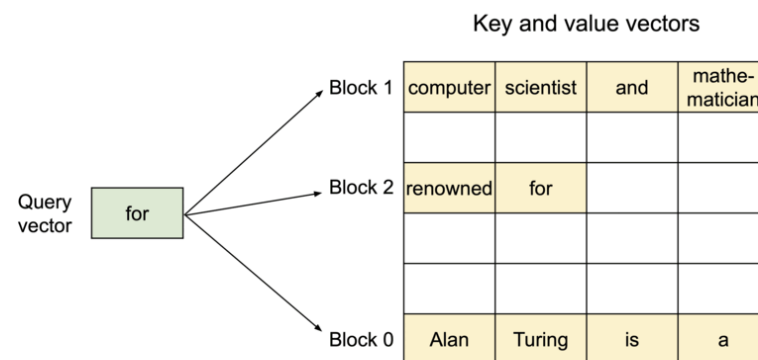
$q \begin{bmatrix} K_1^T & K_2^T \end{bmatrix}$
 $\text{softmax}([A_1, A_2]) = [\alpha \text{softmax}(A_1), \beta \text{softmax}(A_2)]$
 $\text{softmax}([A_1, A_2]) \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \alpha \text{softmax}(A_1) * V_1 + \beta \text{softmax}(A_2) * V_2$
 $\begin{bmatrix} a_1 & a_4 \end{bmatrix} \begin{bmatrix} a_5 & a_6 \end{bmatrix} \begin{bmatrix} p_1 v_1 + p_2 v_2 \end{bmatrix}$

$q \begin{bmatrix} |k_1| & |k_2| & |k_p| \end{bmatrix}$
 $q K_1^T = [a_1 \ a_4] \quad q K_2^T = [a_5 \ a_6]$
 $\begin{pmatrix} e^{a_1} \\ e^{a_2} + e^{a_3} + e^{a_4} \end{pmatrix}$



Paged Attention

- Tensor operations require contiguous memory
- How to compute attention *softmax* across fragmented memory?
- Paged Attention!



$$\text{softmax}([A_1, A_2]) = [\alpha \text{softmax}(A_1), \beta \text{softmax}(A_2)]$$

$$\text{softmax}([A_1, A_2]) \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \alpha \text{softmax}(A_1) * V_1 + \beta \text{softmax}(A_2) * V_2$$



How vLLM & Paged Attention results in efficient inference?

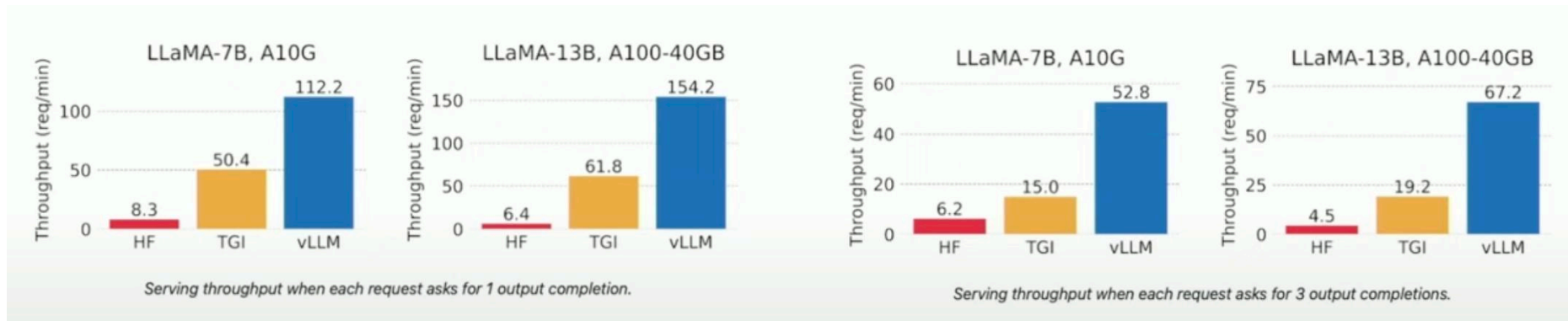
Reduce memory fragmentation with paging ✓

Further reduce memory usage with sharing ✓

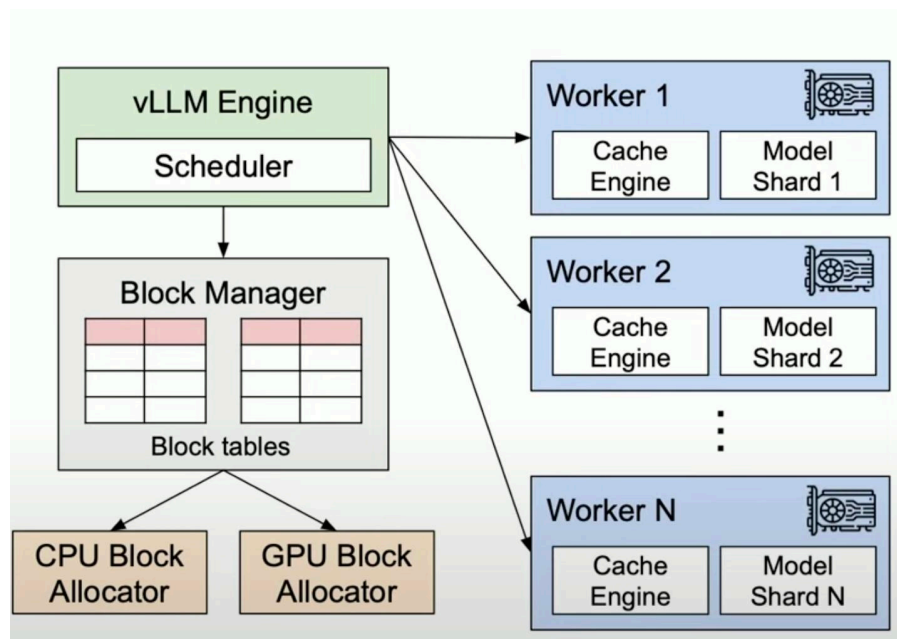


Comparison with HuggingFace and TGI (2023)

- Up to **24x** higher throughput than HuggingFace (HF)
- Up to **3.5x** higher throughput than Text Generation Inference (TGI)



System Architecture and Implementation



End to end llm serving engine

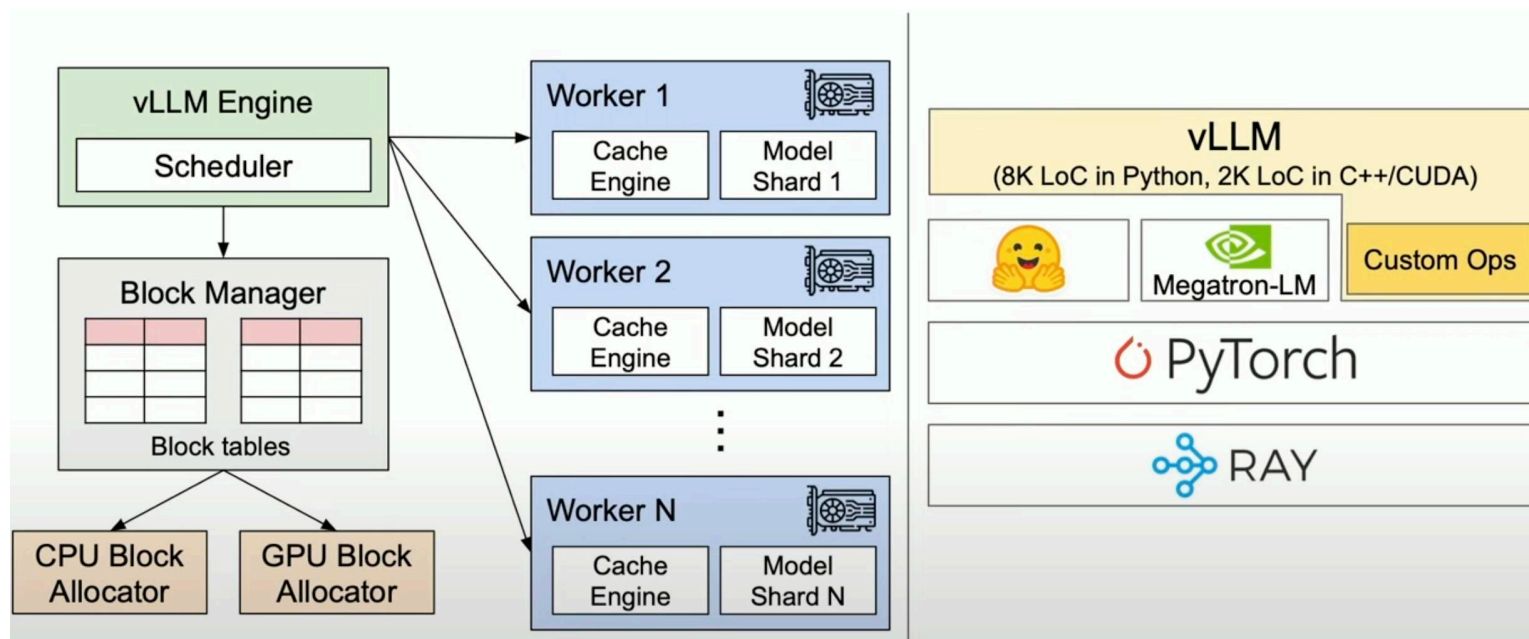
3 components –

- A frontend
- A distributed model executor (TP on single node, PP across nodes)
- A scheduler

Centralized engine to manage block table

- At each iteration, it sends GPU memory requests to the GPUs;
- Cache engine in the GPU allocates the physical memory blocks





Till now...

- **Motivation** – Inference is sequential, memory bound and slow, with high latency
- **KV caching** – avoids re-computation of Keys and Value matrices
- **Paged Attention and vLLM** - efficient memory management
- Can we use **Flash Attention** to speed up decoding?

